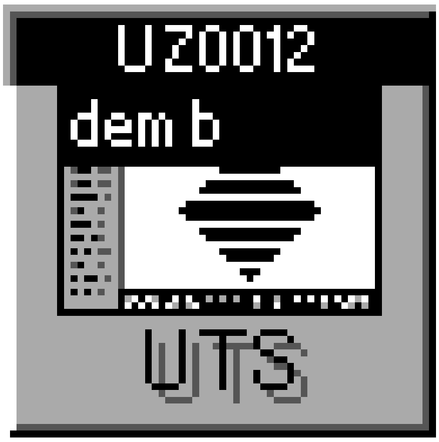
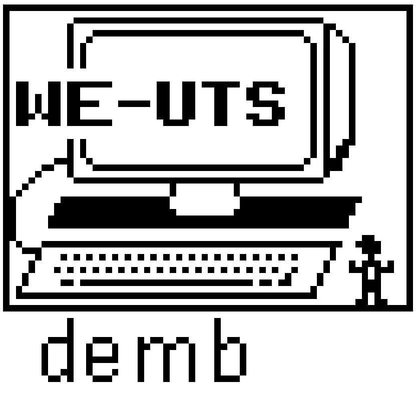
WE-UTS

Installation and
User Reference Manual
Version 1.63x

For NEXTSTEP



and X windows



workstation ag · Rohrstrasse 36/38 · CH-8152 Glattbrugg

The information in this manual is believed to be correct as of the date of publication, however it is subject to change without notice and does not represent a commitment on the part of workstation ag. workstation ag disclaims any warranty of any kind, whether express or implied, as to any matter whatsoever relating to this manual, including without limitation the merchantability or fitness for any direct, indirect, special, incidental or consequential damages arising out of purchase or use of this manual.

The software described in this manual is furnished under a license agreement and may only be installed, used or copied in accordance with the terms of that agreement.

Table of Contents

1 Preface

1.1 Purpose and audience

This manual allows UNIX workstations to interact with UNISYS™ host applications designed for UNISYS™ terminals running in Uniscope mode. This is a user's guide, it is aimed at end users who are UNIX users and are familiar with using the UNISYS™ display terminal and printer. The typical audience is also familiar with UNISYS™ host login and the host applications accessible by the UNISYS™ display terminal product.

1.2 Summary of content

Chapter 2: "Overview"
presents an overview of the WE-UTS terminal emulation package. Included are possible configurations of the software package, supported features, and system software requirements.

Chapter 3: "Installing WE-UTS"
describes the tape contents and installation procedure for WE-UTS.

Chapter 4: "Configuring WE-UTS"
describes the general considerations for configuring workstations and terminals.

Chapter 5: "Considerations about locking configuration file(s)"
describes how to use locking mechanisms for the configuration files.

Chapter 6: "Using WE-UTS terminal emulator"
explains how to use the emulation menus to make your screen customization. This chapter also describes the keyboard layout window, the print layout panel, the macro window and the color mapping window.

Chapter 7: "Keymapper tool"
explains how to use the keyboard mapping utility KM-UTS delivered with your emulation.

Appendix A: "WE-UTS parameter file format"
lists an example WE-UTS parameter file and explains each entry.

Appendix B: "About colors and graphics"
explains how to use colors and graphics with your WE-UTS emulation.

Appendix C: "DCP TELNET example"
shows the relations between the WE-UTS components in a DCP TELNET environment.

Appendix D: "HLC Tp0 example"
shows the relations between the WE-UTS components in an HLC Tp0 environment.

Appendix E: “Status line, window border line, icon and messages”
provides description of the WE-UTS messages areas and their respective content.

Appendix F: “M_func_names” list
provides a list (and brief description) of all WE-UTS function names which can be mapped to the buttons placed around the window (see “Buttons_File” on page 71).

1.3 Conventions used

Throughout this manual we use the following conventions:

The *host system* is the UNISYS™ mainframe computer to which the WE-UTS device emulator communicates through the WE-COMD server.

The *WE-COMD process* is the server used to emulate the UNISYS™ communication protocol.

The *application program* is the job running on the host mainframe to and from which the emulated terminal operator sends and receives data.

1.4 Text fonts used

Throughout this manual fonts are used as follows:

`Typewriter font`
Represents what the system prints on your workstation screen, as well as UNIX system program names and files.

`Boldface typewriter font`
Indicates literal user input, typically commands and responses to prompts that you can type in exactly as printed in the manual.

Boldface font
Emphasized information within the text.

Italic font
Indicates variables or parameters that you replace with an appropriate word or string. Also used for emphasis.

Gray-shaded boxes ░
Contain text that represents interactive sessions. User input is indicated by boldface typewriter font.

Non-shaded boxes □
Contain text that represents listing and non-interactive sessions.

1.5 References

If you need additional informations on any of the major topics mentioned but not explained in this manual, you might try these sources:

UNISYS™ (Sperry) publications

1. UP-9796 Rev x:	UTS System Description
2. UP-9799 Rev x:	UTS System Reference
3. UP-9797 Rev x:	UTS Introduction for Operators
4. UP-9798 Rev x:	UTS Operator's Reference

NeXT manuals

1. User's Reference Manual
2. System and Network Administration

Sun manuals

1. Beginner's Guide to the Sun Workstation
2. System and Network Administration

2 Overview

2.1 Introduction

The WE-UTS product line centers around three products that allow workstations to communicate with UNISYS™ mainframes over TCP/IP Ethernet links (TELNET or Tp0 connections), UDLC or X.25 public or private networks:

#	WE-UTSc	Terminal emulation without graphics
#	WE-UTSg	Terminal emulation with graphics
#	WE-COMD	Communication protocol server

These products provide interactive access to the many applications that run on UNISYS™ mainframes without modifying the mainframe software.

2.2 Configurations

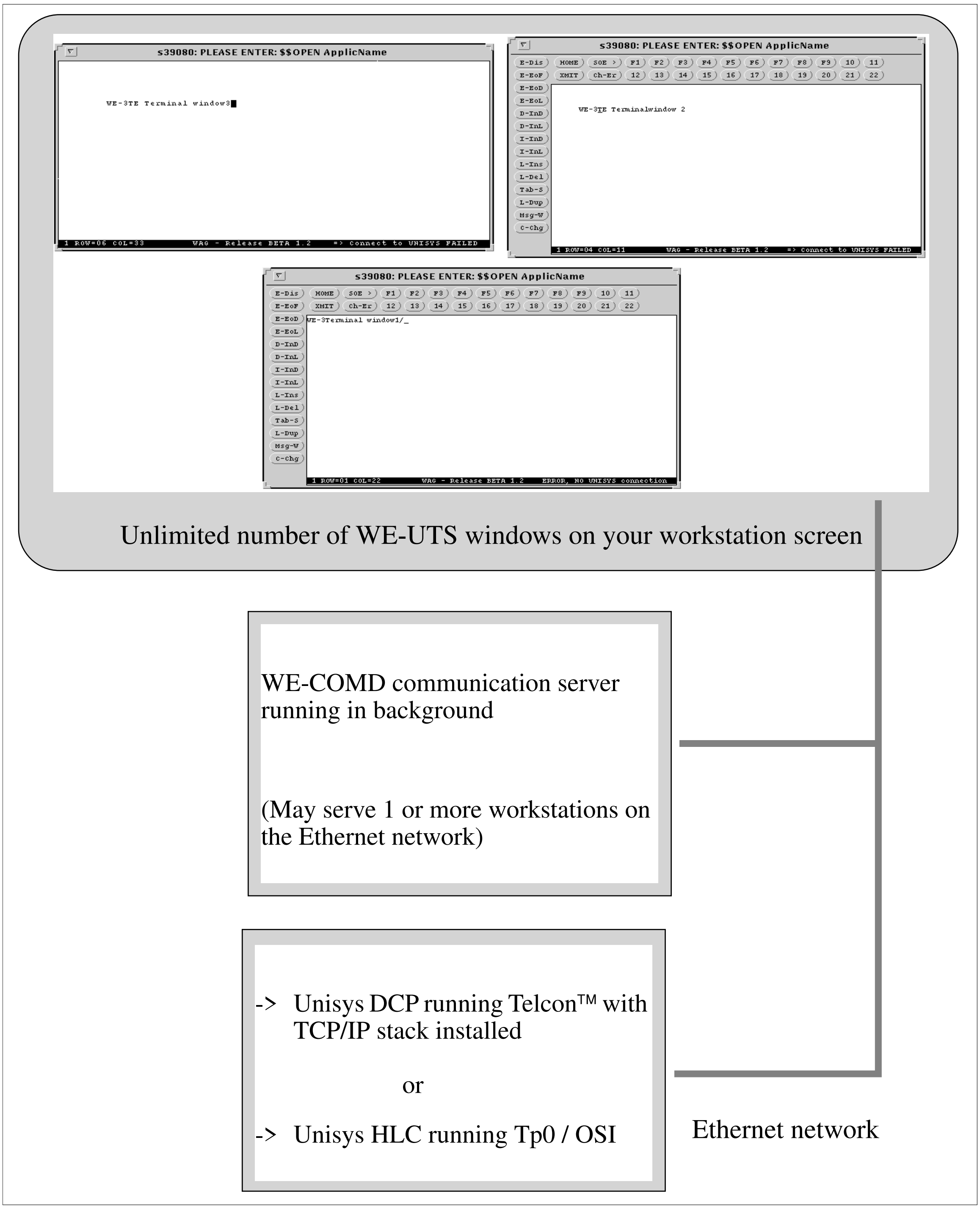
The products are software packages configured to run directly on workstation processors. The software for each product has a gateway component and a workstation component. When a WE-COMD gateway component is installed on a UNIX processor, that processor may become a protocol server for the entire network. The gateway may be shared by any number of workstations, each running any number of WE-UTS terminal windows.

Each of these WE-UTS products isolates the communication protocol support to their respective gateways, thereby buffering the workstations from the complexities and additional overhead of DCA Ethernet (DCP), Tp0 Ethernet (HLC), UDLC or X.25 protocols.

The workstation talks to the gateway using TCP/IP, which provides flexible and high-performance distribution of the WE-UTS terminal sessions.

2.2.1 Typical examples

The following figure highlights the software components of the WE-UTS products.



```
s39080: PLEASE ENTER: $$OPEN ApplicName

      WE-3TE Terminal window3

1 ROW=06 COL=33          WAG - Release BETA 1.2     => Connect to UNISYS FAILED
```

```
s39080: PLEASE ENTER: $$OPEN ApplicName

      WE-3TE Terminalwindow 2

1 ROW=04 COL=11          WAG - Release BETA 1.2     => Connect to UNISYS FAILED
```

```
s39080: PLEASE ENTER: $$OPEN ApplicName
WE-3Terminal window1/_

1 ROW=01 COL=22          WAG - Release BETA 1.2     ERROR, NO UNISYS connection
```

Unlimited number of WE-UTS windows on your workstation screen

WE-COMD communication server running in background

(May serve 1 or more workstations on the Ethernet network)

-> Unisys DCP running Telcon™ with TCP/IP stack installed

or

-> Unisys HLC running Tp0 / OSI

Ethernet network

WE-UTS emulation components for Ethernet connection

2.3 Feature summary

Workstation WE-UTS terminal emulation offers support for applications written for U100, U200, UTS 400, UTS 40, UTS 60 color graphic (WE-UTSg only) and SVT UNISYS™ terminal families.

2.3.1 WE-UTSc basic datastream operations

All operations accepted by the <basic> Unisys UTS 30 like:

Cursor positioning operations
Field control characters (FCC's and extended FCC's)
Control page operations
Keyboard FCC generation
All attribute combinations (including blink)
Peripheral control, etc.

2.3.2 WE-UTSc extended datastream operations

Color FCC's (a UTS 60 feature)

80/132 columns support (a SVT feature)

2.3.3 WE-UTSg datastream operations

All operations accepted by the WE-UTSc.

Graphic orders in the UTS 60 format.

2.3.4 WE-UTS keyboard support

All functions of the WE-UTS keyboard are implemented directly on the workstation keyboard (configurable through an interactive keyboard mapper utility) or on the screen area around the terminal window (clickable with the mouse).

-> Limitations against Unisys UTS 30

Peripheral operations are limited to the printer. Diskettes, magnetic stripes reader and the like are not supported.

The WE-UTS emulation doesn't provide programmable character set like the <programmable> Unisys UTS 30.

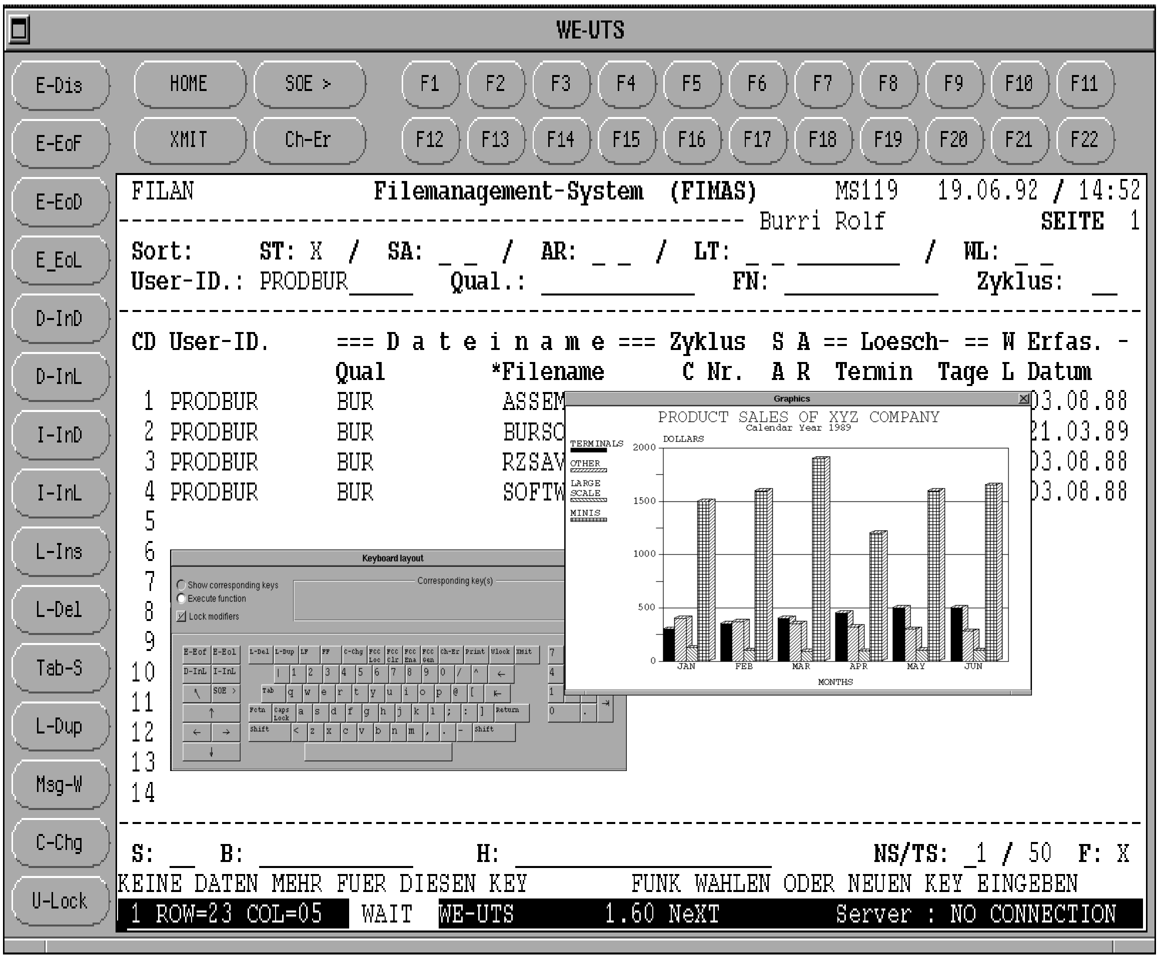
2.3.5 Additional features not provided by the UNISYS™ terminals

– Unlimited number of simultaneous WE-UTS sessions (windows) on a single workstation. Possibility to save screen space by iconifying a window without interruption of the terminal session.

– Cut and paste between WE-UTS windows and other windows.

– Variable scroll speed.

- Context sensitive help for all keys and parameters.
- Keyboard mapping and up to 12 macro functions can be defined separately for each WE-UTS window.
- Cursor positioning either by key depression (i.e.: tabs) or by mouse clicking anywhere in the WE-UTS window.
- A graphical representation of an original Unisys UTS keyboard can be displayed at any time on the screen and used with the mouse.
- Data destined to the printer (screen hardcopies or data coming directly from the host) are directed either to a file or to the printer spooler. This means that the host will never have to wait for printer readiness, paper low, etc.



```
FILAN            Filemanagement-System  (FIMAS)      MS119   19.06.92 / 14:52
------------------------------------------------ Burri Rolf           SEITE  1
Sort:     ST: X  /  SA: _ _  /  AR: _ _  /  LT: _ _ ________  /  WL: _ _
User-ID.: PRODBUR_____   Qual.: ____________   FN: ____________   Zyklus:  __
------------------------------------------------------------------------------
CD User-ID.     === D a t e i n a m e === Zyklus  S A == Loesch- == W Erfas. -
                Qual        *Filename      C Nr.  A R  Termin  Tage L Datum
 1 PRODBUR      BUR          ASSEM                                   03.08.88
 2 PRODBUR      BUR          BURSC                                   21.03.89
 3 PRODBUR      BUR          RZSAV                                   03.08.88
 4 PRODBUR      BUR          SOFTW                                   03.08.88
 5
 6
 7
 8
 9
10
11
12
13
14
------------------------------------------------------------------------------
S: __   B: ____________      H: ______________________      NS/TS: _1 / 50  F: X
KEINE DATEN MEHR FUER DIESEN KEY        FUNK WAHLEN ODER NEUEN KEY EINGEBEN
 1 ROW=23 COL=05    WAIT   WE-UTS        1.60 NeXT        Server : NO CONNECTION
```

○ Show corresponding keys
○ Execute function
☑ Lock modifiers

An example of a WE-UTS terminal emulation window

3 Installing WE-UTS

3.1 DCA Ethernet host connection using the “WE-COMD” communication server

The installation procedure consists of 4 consecutive steps as follows:

a) Extracting the product from the distribution media.

b) Adding entries to your <etc/hosts> and <etc/services> files to declare the different Internet names and TCP ports used for the communication between the WE-UTS components and the UNISYS™ host (DCP).

c) Creating configuration files which will be used by the WE-UTS terminal emulation to configure itself and address the proper UNISYS™ application for the sessions you want to establish.

d) Installing the <WE-UTS.info> file for On-line help.

3.1.1 Extracting the product from the distribution media

The procedure for extracting the products from the distribution media may vary depending on your particular workstation and/or environment. The actual procedure is explained on a sheet attached with your media. Please read this paper and follow the instructions contained therein.

When the installation is complete, you should find the following files in the installation directory:

`README`	The latest information about WE-UTS
we-uts	The terminal emulator
we-comd	The communication server
km-uts	The keymapping tool
Sample.UTSg_config	A sample configuration file for UTSg (graphic)
Sample.UTSc_config	A sample configuration file for UTSc (text only)
keylay.dat	An ASCII image of the UTS terminal keyboard
color.dat	A sample color file
macro.dat	A sample macro file
button.dat	A sample button file

If you purchased WE-UTS for an X11 server machine, the following supplemental files will be provided:

ps_generic.dat	Used to generate PostScript hardcopies
cour.pfa	Scalable font used for graphics (standard)
courb.pfa	Scalable font used for graphics (bold)
couri.pfa	Scalable font used for graphics (italic)

3.1.2 Adding <hosts> and <services> entries for WE-UTS

According to your own network configuration you will have to add correct entries into /etc/hosts and /etc/services (may be on your YP/NIS or Netinfo server) to allow the WE-COMD and the WE-UTS programs to exchange informations between each other and with the <Unisys HLC or DCP> communication controller.

Appendix C and D are examples for the relationships between the emulation components for both DCP and HLC communication paths. Thereafter, you will find some more indications about the important points:

– The <WE-COMD> and <WE-UTS> components need a TCP port to communicate together. You can assign any free port number and service name for that purpose. For example, add the following entry into </etc/services>:

```
we-comd_serv        1030/tcp
```

– The <WE-COMD> and the <Unisys HLC or DCP> communication controller need a TCP port to communicate together. You can choose the service name freely, but **the TCP port MUST be 256 for a TELNET DCP connection and 102 (iso_tsap) for a Tp0 HLC connection**. This is imposed by Unisys and ISO conventions. For example, add the following entry into </etc/services>:

```
For DCP (TELNET):     we-dcp_serv        256/tcp
For HLC (Tp0):        we-hlc_serv        102/tcp
```

– Finally, you must ensure that the workstations on which the WE-UTS emulation components will be running as well as the Unisys DCP and/or HLC are declared in the </etc/hosts> files.

3.1.3 Creating configuration file(s) for WE-UTS

For a complete description of all possible entries in a parameter file, please refer to Appendix A on page 61. Thereafter, we will discuss where and how WE-UTS searches for parameter file(s) and some general rules valid for all entries in these files.

Format of an entry in a parameter file

All entries have the format:

ParameterName *Value* *Comment*

```
i.e.: Comd_Host     cracker      # my workstation
```

If you don’t want to assign a new value to a parameter, you **must** remove the corresponding entry from the parameter file. A line like

```
Comd_Host
```

with no *Value* would set `Comd_Host` to **nothing**. This would, of course, not work ...

Up to four parameter files are possible

When started, WE-UTS looks sequentially for 4 parameter files (all have the format specified in Appendix A on page 61).

– First, it looks for a file with the name **<.we-uts-rc>** in the **</etc/we-uts-config>** directory. If it is found, the parameters contained are extracted and validated.

– Secondly, it looks for a file with the name **<.we-uts-rc>** in the user’s home directory. If it is found, the parameters contained are extracted and validated.

– Thirdly, the <command line> is searched for a **<-sf *FileName*>** argument. If one is present, the parameters contained are extracted and validated.

– Fourthly, the <command line> is searched for a **<*FileName*>** entry as the first argument. If one is present, the parameters contained are extracted and validated.

NOTE 1: There is no obligation to have (or use) all 4 parameter files. This is provided to give you more flexibility to divide between common parameters for all WE-UTS sessions, less common entries valid for a group of sessions and finally parameters valid for a unique session only (i.e.: Term_Name).

NOTE 2: If any parameter is defined in more than one of the four possible files, the value read in the last file will be applied to that parameter. This allows you to apply your own default values for some parameters.

3.1.4 Installing the <WE-UTS.info> file for On-line help

The <WE-UTS.info> file provides help information for WE-UTS. You may use it on any X window system. Here is how WE-UTS searches for this file:

- If HELPPATH is set in your environment, you merely need to copy the <WE-UTS.info> file into the directory indicated by HELPPATH.

- If HELPPATH is not set in your environment, you may either:

  - Set HELPPATH and copy the <WE-UTS.info> file there.
  - Copy the <WE-UTS.info> file into </usr/lib/help> (default).

NOTE: There is currently **no** help available for the NEXTSTEP version.

4 Configuring WE-UTS

After you have created the entries in the UNIX “/etc/services” and “/etc/hosts” files and edited your configuration file(s), it’s now time to try your first connection to the Unisys host. In fact, the Unisys device we will connect to is not the host itself, but it’s DCP (Data Communication Processor) or HLC (Host Line Controller) component which is connected to the Ethernet network.

WE-UTS will be started in 5 phases as outlined below

1. *Start the WE-COMD component (communication server)*
   This is done once on each workstation (sometimes once for the whole network). For that reason, starting of the WE-COMD server is mostly done automatically in the <rc.local> shell script.

2. *Start the “WE-UTS” uniscope terminal emulator*
   Is done upon request of the user. You can start as many WE-UTS’s as you want on a single workstation, as long as you have a valid license that covers that many concurrent windows.

3. *1st (automatic) connection to the host*
   Occurs only if all parameters necessary for a connection were found in the configuration file(s).

4. *Use the $$OPEN and $$CLOSE commands*
   Can occur as often as you want.

5. *Start “we-uts_host” simulator*
   This is only for a demonstration of the WE-UTS terminal emulator if you have no Unisys connection available.

4.1 Starting the “WE-COMD” communication server component

The syntax for starting the WE-COMD component is as follows:

we-comd ServiceName [-sm] [-lc] [-t type] [-lp LogFilePath]

1. *<ServiceName>*
   Must be the first parameter. This service will be used for communication between <we-comd> and <we-uts>. His name must be the same as the <Comd_Service> entry in the WE-UTS parameter file. It must also be added to the </etc/services> file (or YP, Netinfo).

2. *<-sm>*
   Optional (means short messages and is optional) will restrict “WE-COMD” to print less detailed information messages.

3. *<-lc>*
   Optional (means log to console) WE-COMD write all the messages to the console. Normally, WE-COMD creates a file (see “<-lp LogFilePath>” on page 20)

   </tmp/we-comd.PID> where PID is the process ID

at start-up time. If errors occur, messages will be written into that file. If you start WE-COMD with the *[lc]* option, no error logging file will be opened and the messages will be sent to the console instead.

REMARK: Depending on your particular system, writing messages on the console may destroy the appearance of your graphical environment. Mostly, this can be repaired with a <refresh> function of your window manager.

4. *<-t type>*
   Where type may be one of the following:

telnet	(used for communication to a DCP)
tp0	(used for communication to a HLC)

REMARK: If the *[-t type]* optional parameter is not given, it is defaulted to *[-t telnet]*. This only works for a DCP TELNET connection.

5. *<-lp LogFilePath>*

   Where LogFilePath may be any existing directory where you may create new files.

   Example:

```
machine% we-comd we-comd_serv -t tp0 -lp /Log
```

When WE-COMD starts, it will print a message like:

```
<we-comd 3.24 pid:4279> at port <1030> in mode <tp0>
<we-comd 3.24> error logging will be on </Log/we-comd.4279>
```

This means that WE-COMD is running and is waiting for WE-UTS connections at TCP/IP port 1030 (the port you have assigned for “Comd_Service”). Afterwards, WE-COMD does its job in the background and you will get **no** messages unless any special condition occurs. Error logging will occur on file </Log/we-comd.4279>.

If the <ServiceName> obligatory parameter is missing or wrong you will get either of the following messages:

For example, if you type

```
we-comd
```

you get the following message:

```
we-comd: Correct syntax is: we-comd ServiceName
          [-sm] [-lc] [-t type] [-lp LogFilePath]
```

If you type

```
we-comd far_west
```

and service <far_west> doesn’t exist, you will get the following message:

```
we-comd: far_west not a server
```

4.1.1 Hints for starting the “WE-COMD” server from the “rc” file

Under UNIX, most daemons are started at the end of the boot procedure. WE-COMD is no exception and may also be started there. All parameters described above remain valid. Depending on your particular system, you may add the following line into </etc/rc>, </etc/rc.local>, etc.

```
we-comd comd-serv -sm -t tp0 >> /tmp/we-comd.startup 2>&1
```

This would start <we-comd> in <tp0> mode and redirect any messages occurring at start-up time to the file </tmp/we-comd.startup>. Redirecting start-up messages to a file may be useful for debugging purposes.

4.2 Starting the “WE-UTS” terminal emulator component(s)

The syntax for starting the “WE-UTS” component is as follows:

we-uts [*FileName*][**-sf** *FileName*]

1. <‘*FileName*’> File name of configuration file for WE-UTS

2. <-sf ‘*FileName*’> Special configuration file for WE-UTS

All the configuration files have the same format (see “Creating configuration file(s) for WE-UTS” on page 17).

Example:

```
machine% we-uts /usr/we-uts/john/myconfig1
                -sf /usr/we-uts/all/grpconfig
```

Then, “WE-UTS” will try to read (in sequence):

– A file named <.we-uts-rc> in the </etc/we-uts-config> directory.

– A file named <.we-uts-rc> in your home directory.

– A file named <grpconfig> in the </usr/we-uts/all> directory.

– A file named <myconfig1> in the </usr/we-uts/john> directory.

NOTE 1: All the 4 possible parameter files have the same format which is detailed under Appendix A on page 61.

NOTE 2: If any mandatory parameter is missing, you will get an error message in the form

```
WE-UTS missing parameter:
- Comd_Host
- Mainframe_Service
```

detailing the missing entry(ies).

NOTE 3: If all mandatory parameters were present and valid you will get a WE-UTS window on your workstation screen.

NOTE 4: A connection to the host will be attempted if

- the <Appl_Name> optional parameter was found **and**
- the <Mainframe_Host> optional parameter was found.

Otherwise, you will get the message

```
Please, enter $$OPEN ...
```

in the upper window border.

NOTE 5: If any parameter was invalid or the necessary fonts were not found or any error occurred at start time, you will **not** get any WE-UTS window but one of the error messages.

4.2.1 Error messages while starting the “WE-UTS” terminal emulator

The following are fatal errors which can occur when starting the WE-UTS terminal emulator component. When they occur, **no** WE-UTS window will be created and you will have to correct the error(s) before proceeding.

a) If obligatory parameter(s) is (are) missing, you will get a message like:

```
WE-UTS missing parameter(s):
- Mainframe_Service
- Term_Type
```

Detailing all missing (obligatory) parameters.

Action => Add the corresponding parameters in a configuration file.

b) If the <Comd_Host> from the configuration file(s) can’t be found, you will get the following message:

```
WE-UTS:Comd_Host = <HostName> NOT found
```

Where: <HostName> is the name found in the configuration file(s).

Action => Correct the <Comd_Host> entry in the configuration file so it matches a valid entry in </etc/hosts> (YP or Netinfo).

c) If the <Comd_Service> from the configuration file(s) can’t be found, you will get the following message:

```
WE-UTS:Comd_Service = <ServiceName> NOT found
```

Where: <ServiceName> is the name found in the configuration file(s).

Action => Correct the <Comd_Service> entry in the configuration file so it matches a validentry in </etc/services> (YP or Netinfo).

d) If the protocol server <WE-COMD> is not running at all or a connection to it is impossible, you will get the message:

```
WE-UTS:init_socket:connect: Connection refused
```

Action => Start the <WE-COMD> component on the <Comd_Host> machine under the <Comd_Service> declared in the configuration file(s). Consult previous chapters.

e) If any of the font parameters from the configuration file(s) can't be found, you will get the following message:

```
WE-UTS:Font <FontName> NOT found
```

Where: `<FontName>` is the font name found in the configuration file(s).

Action => Specify fonts accessible to your X server. Don't forget that only fixed point fonts are usable for this terminal emulation.

4.2.2 Note for NeXT users

Under NEXTSTEP, you may start an emulation by double-clicking on an icon corresponding to a configuration file. This is possible only if the two following conditions are met:

– The application (we-uts) is installed in a directory known by the Workspace Manager like:

/Apps

~/Apps

/LocalApps

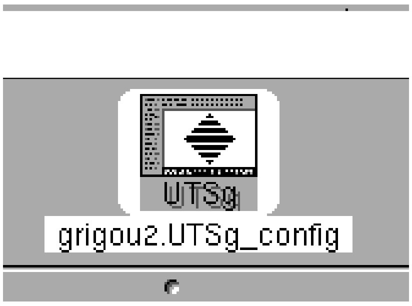
...

– The configuration file you want to click on have the extension:

.UTSg_config (if you purchased a WE-UTSg <UTS 60>) emulation

.UTSc_config (if you purchased a WE-UTSc <UTS 40>) emulation

If both conditions are met, the <xxx.UTSc_config> file will appear in your browser with the same icon as the <we-uts> application.



REMARK: During emulation start-up, some of the error messages described in the previous chapter may be replaced by panels for better readability. Other messages may be sent to the system console. Therefore, it is recommended to have a console window open during the emulation installation phase.

When started by double-clicking, <WE-UTS> will still try to read

– A file named <.we-uts-rc> in the </etc/we-uts-config> directory
– A file named <.we-uts-rc> in your home directory

to get configuration information. The configuration file you click on will be read last and parameters contained herein may overwrite previous defined values.

4.3 Establishing a WE-UTS to host connection

A connection to the host can be established:

a) Automatically at WE-UTS start-up if both <Appl_Name> and <Mainframe_Host> were provided in the configuration file(s).

b) At any time when the upper window border displays the message:

```
Please, Enter $$OPEN Appl_Name, [Unisys_Host]
```

This invitation will be displayed when:

Just after WE-UTS start-up if either or both of <Appl_Name> and <Mainframe_Host> are missing.

At any time after the current session has been closed with a $$CLOSE or by any other mean (time-out, error, etc.).

4.4 The $$OPEN and $$CLOSE commands

Like an original Unisys terminal, WE-UTS can establish and cancel a session with the $$OPEN and $$CLOSE commands. This command is working the same way for DCP or HLC connections (only the format is different).

4.4.1 $$OPEN

The $$OPEN command can be given at any time and has the following format:

$$OPEN ApplName[,SystName] (for DCP <telnet>)

$$OPEN ApplName[,ApplServer][,SystName] (for HLC <tp0>)

Where:

`ApplName` is the name of the application you want to be connected to (i.e.: demand, tip, etc.).

`ApplServer` is the name of the application server you want to be connected to (i.e.: TIPCSU, RSDCSU, etc.). This parameter may be given on a Tp0 connection only (HLC).

`SystName` is the name of the Unisys DCP or HLC you are connected to (as entered in the UNIX file </etc/hosts>).

NOTE 1: The $$OPEN command is not case sensitive (i.e.: $$OpEn is valid).

NOTE 2: The $$OPEN must be the only field transmitted when you depress XMIT.

For that reason, you must type it at the home position or immediately preceded by a SOE character. Furthermore, the cursor **must** stand just behind the last character of your command when you depress the XMIT key.

NOTE 3: If no parameters are given, $$OPEN try to connect to the preceding application (the one you were connected to before you made a $$CLOSE or an error occurred). If you give only the ApplName, the connection will be done using the <old> ApplServer (Tp0 connections only) and the <old> SystName.

NOTE 4: If some parameters are missing or wrong, you will get an error message in the status line communication area (see Appendix E on page 85).

NOTE 5: If a session is already active when you give the $$OPEN command, it will be automatically closed before opening a new one with the new parameters.

NOTE 6: If WE-COMD is started with the <- sm> (short messages) option, you will **not** be prompted for a SystName and will **not** be able to change it with a $$OPEN order.

4.4.2 $$CLOSE

The $$CLOSE command can be given at any time while a session is active and has the following format:

$$CLOSE

NOTE 1: The $$CLOSE command is not case sensitive (i.e.: $$cLoSe is valid).

NOTE 2: The $$CLOSE must be the only field transmitted when you depress XMIT.

For that reason, you must type it at the home position or immediately preceded by a SOE character. Furthermore, the cursor **must** stand just behind the last character of your command when you depress the XMIT key.

NOTE 3: When the $$CLOSE command has completed its job, you will get an appropriate message in the status line communication area (see Appendix E on page 85).

4.5 Starting the “we-uts_host” simulator

This is only for a demonstration of the WE-UTS terminal emulator if you have no Unisys connection available. This may also be used to check the correctness of your installation independently of any Unisys connection.

In the subdirectory <demo>, you will find the simulator program <we-uts_host>. When you are done with the installation and configuration of the WE-UTS product, you may start it for a test with the following command:

we-uts_host *SymFileName MainframeService Type*

1. *SymFileName*
   File name of the datafile to be used by the <we-uts_host> program. You can choose any of the <.sym> files found in the <demo> subdirectory.

2. *MainframeService*
   The name of the mainframe service it must be the same name as in the parameter Mainframe_Service of the <we-uts-config> and </etc/services> files.

3. *Type*
   Will define the type of connection between WE-COMD and WE-UTS_HOST. This **must** match with <type> chosen for the WE-COMD server component. If **no** <type> is given, it is defaulted to <telnet>. Possible values for type are:

   telnet
   tp0

   Examples:

   ```
   machine% we-uts_host MigDem1128.sym tp0
   ```

   will start the simulator with the demo file <MigDem1128.sym> in <tp0> communication mode.

   ```
   machine% we-uts_host MigDem1128.sym
   ```

   will start the simulator with the demo file <MigDem1128.sym> in <telnet> communication mode (default).

REMARK: Please look at the README file of the <demo> subdirectory for the latest informations about <we-uts_host>.

5 Hints about locking configuration file(s)

If you intend to run many WE-UTS window on a single or on multiple workstations, you need control on the usage of some connection parameters.

The most important fact is, that at any given time, one and only one user may be connected using a given terminal identification <Term_Name>.

For that reason, we have provided the <Lock_Setup> parameter in your configuration file(s). The configuration file, which will be locked if you set this option (set it to 1), is the one given without <-sf> as the 1st argument on the command line.

Therefore, we recommend that you put the <Term_Name> in that configuration file and use <Lock_Setup = 1>. If you do so, WE-UTS will lock that file at start-up time for the whole duration of the Uniscope session. If you try to start a second window with the same parameter file, you will get a message telling you, that the configuration file is locked and no window will be created. This also provides the possibility to create a configuration file management software examining which configuration(s) is (are) currently in use.

REMARK: The <Lock_Setup> option can only be used if you have <write> access to the concerned configuration file.

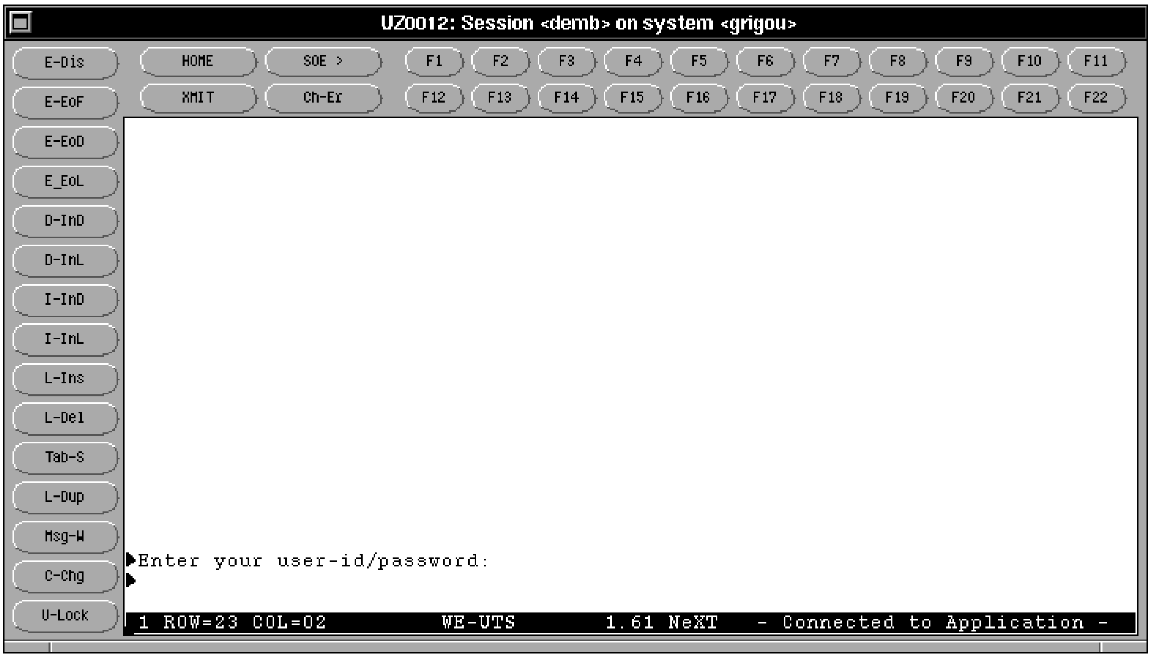
6 Using WE-UTS terminal emulator

6.1 Foreword

This chapter applies to the X and NEXTSTEP version. When nearly identical, only an X or a NEXTSTEP picture is shown in this manual. When differences exist, the X picture is shown on the left and the NEXTSTEP picture on the right.

This chapter introduces you to the terminal emulator.

If you start the WE-UTS and the host connection is successful, the following window appears (data will probably be different):

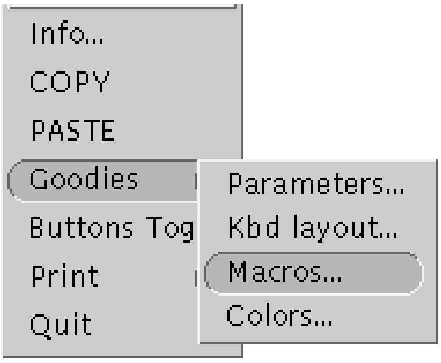


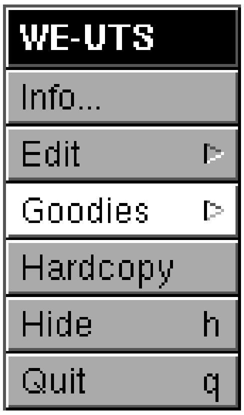
```
▶Enter your user-id/password:
▶
 1 ROW=23 COL=02          WE-UTS        1.61 NeXT    - Connected to Application -
```

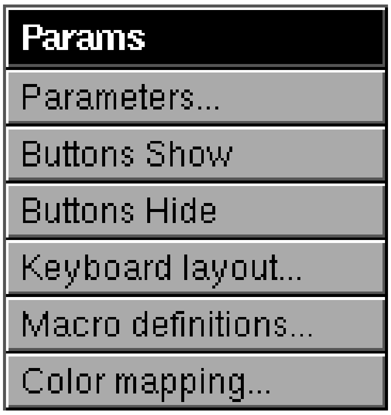
6.2 Main menu

The two pictures on page 30 show the Main menus for X and NEXTSTEP. Under X, you will get it by clicking the right mouse button, while the cursor is in the emulator main window. For NEXTSTEP, it will appear on the top left of your screen while the emulation window is selected.

Activating “Quit” leaves the terminal emulator.

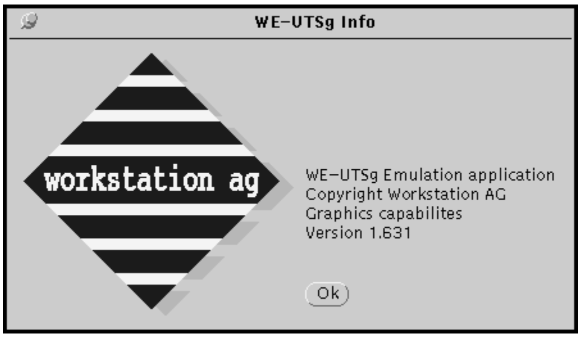






6.2.1 Info

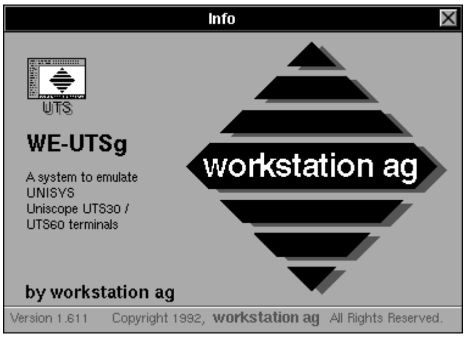
Will display an information panel containing the software release, copyright information, etc.



WE-UTSg Info

WE-UTSg Emulation application
Copyright Workstation AG
Graphics capabilites
Version 1.631

Ok



Info

WE-UTSg

A system to emulate
UNISYS
Uniscope UTS30 /
UTS60 terminals

by workstation ag

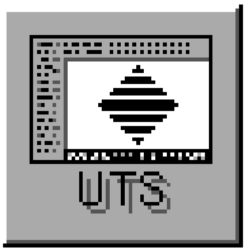
Version 1.611   Copyright 1992, workstation ag All Rights Reserved.

6.2.2 Copy, Paste

Will allow you to Copy and Paste between emulator windows and other applications. The way you can use Copy and Paste in the emulation is the same as for any other application. Under NEXTSTEP, Copy and Paste are in the Edit submenu.

6.2.3 Hide (NEXTSTEP only)

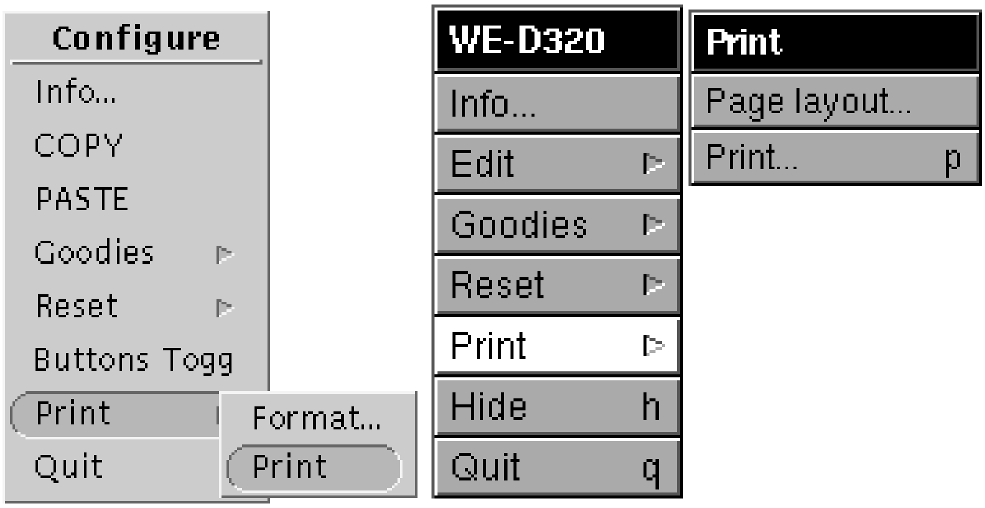
If clicked, any window or panel as well as the Main menu panel will be removed from your screen and only the UTS icon will remain.



We recommend **not** to use the Hide menu function but to iconify the emulation main window by clicking in the upper left button (see Appendix E on page 85).

6.2.4 Print

When you choose this option, the following submenu appears:



Print and print graphics

Will make a hardcopy of the text or graphic (WE-UTSg only) window respectively. The format of the printout is defined by:

-> Values set by the configuration file(s) read at WE-UTS start-up (see Appendix A on page 61).

-> Values entered at any time using the format or page layout panel.

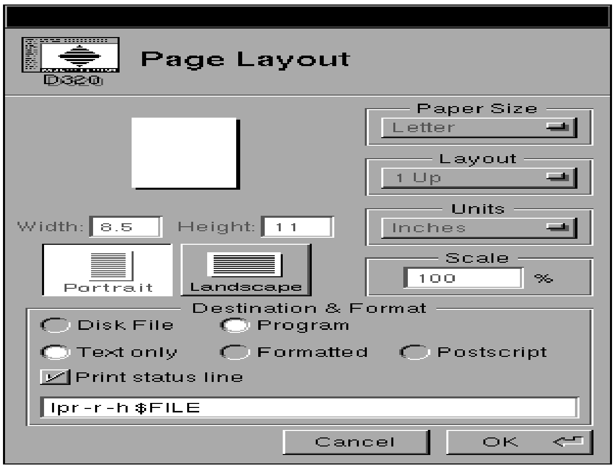
Format (X) **or page layout** (NEXTSTEP)

Foreword for NeXT users:

If you choose the PostScript “Print_Format” and make hardcopies, the standard NeXT print panel will appear each time as with all NeXT applications. With that panel, you will be able to print or save your print at will.

This means that the destination (disk file or program) and the text field at the bottom of the page layout panel entries are **useless for NeXT PostScript hardcopies**. However, they still **apply to data coming directly from the host** (i.e.: ADS print) because such data are never printed in PostScript.

Clicking on format (X) or page layout (NEXTSTEP) will show one of the following panels:



Page Layout

Paper Size: Letter

Layout: 1 Up

Units: Inches

Width: 8.5 Height: 11

Scale: 100 %

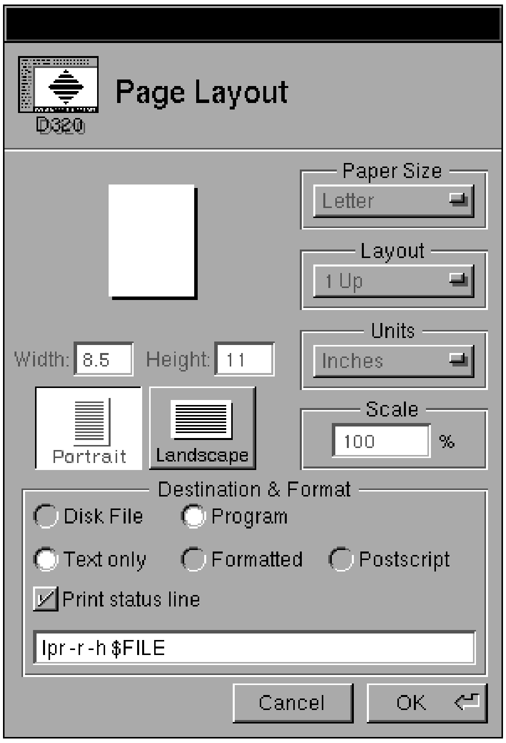
Destination & Format

○ Disk File ○ Program

○ Text only ○ Formatted ○ Postscript

☑ Print status line

lpr -r -h $FILE



Page Layout

Paper Size: Letter

Layout: 1 Up

Units: Inches

Width: 8.5 Height: 11

Scale: 100 %

Destination & Format

○ Disk File ○ Program

○ Text only ○ Formatted ○ Postscript

☑ Print status line

lpr -r -h $FILE

At program start-up, these panels contain defaults or values set by the corresponding config (parameter) file(s) entries (see Appendix A on page 61).

Although the panels for X based machines and for NEXTSTEP looks very different. Buttons and text fields have the same function. Thereafter, we will discuss each of them:

Destination

-> Disk

WE-UTS will create disk files containing the print data in the given format. The text field at the panel bottom contains the file name. There are two cases:

- If you don’t use $ext, the file name given will be unique and therefore be overwritten by each hardcopy (or data chunk coming from the host).
- If you put $ext at the end of your file name, new files will be created automatically for you by each hardcopy (or data chunk coming from the host). The following extensions will be added automatically by WE-UTS.

If “Format” is Text only:

ProcessNumber-SequenceNumber-Kind

If “Format” is Formatted:

ProcessNumber-SequenceNumber-Kind.fmt

If “Format” is PostScript:

ProcessNumber-SequenceNumber-Kind.ps

Where:

– *ProcessNumber* is the UNIX allotted process number of the WE-UTS process.

– *SequenceNumber* is a number (starting at 0 and incremented by 1 at each print).

– *Kind* may be:

hcopy (for hardcopies)
int1 (for host prints with @@prnt/@@nopr)
normal (for host ADS prints with DC2)
transp (for host ADS prints with ESC/DC2)
form (for host ADS prints with ESC/H)

CAUTION: If you edit the text field to change the file name, don’t forget to **press return** to validate the typed text.

REMARK: Defaults come from the following configuration file entries:

Print_Destination	*Disk*
Disk_Print	*~/MyPrints.$ext*

-> Program

WE-UTS will create temporary disk files (/tmp/we-uts/$ext) containing the print data in the given format for the meaning of $ext (see “Destination” on page 32). The text field at the panel bottom contains the name of the program to which WE-UTS will pass this temporary file for further handling. A typical entry in this text field may be:

lpr -r -h $File

Where:

lpr	is the name of the program to be called.
-r -h	are parameters passed to program.
$File	token which will be replaced by WE-UTS with the name of the file just created.

CAUTION: If you edit the text field to change the program name, don’t forget to **press return** to validate the typed text.

REMARK: Defaults come from the following configuration file entries:

Print_Destination	*Program*
Disk_Print	*lpr -r -h $File*

Format

-> Text only

WE-UTS will create hardcopies containing exclusively standard ASCII characters. This means that attributes and foreign characters will not be rendered.

-> Formatted

Currently the same as Text only.

-> PostScript

WE-UTS will create hardcopies which will be an exact image of the current text or graphic window (depending if you click print or print graphics). If Print_Status_Line is **on**, the status line and a window border will be printed as well.

NOTE: Printouts of the graphic window (WE-UTSg only) may only be done in PostScript format.

REMARK: Defaults come from the following configuration file entries:

Print_Format	*Text*
	Formatted
	PostScript

Orientation, paper size and scale

These entries are valid only for PostScript format. They are used by WE-UTS to generate the proper PostScript code according to the paper size, orientation and scale chosen.

REMARK: Defaults come from the following configuration file entries:

Print_Output_Size	*Letter*
	A5
	A4
	A3
Print_Orientation	*Portrait*
	Landscape
Print_Scale	*Any number between 30 and 300*

Print_Status_Line

If Print_Status_Line is **on** (set), the status line content will be added to hardcopies.

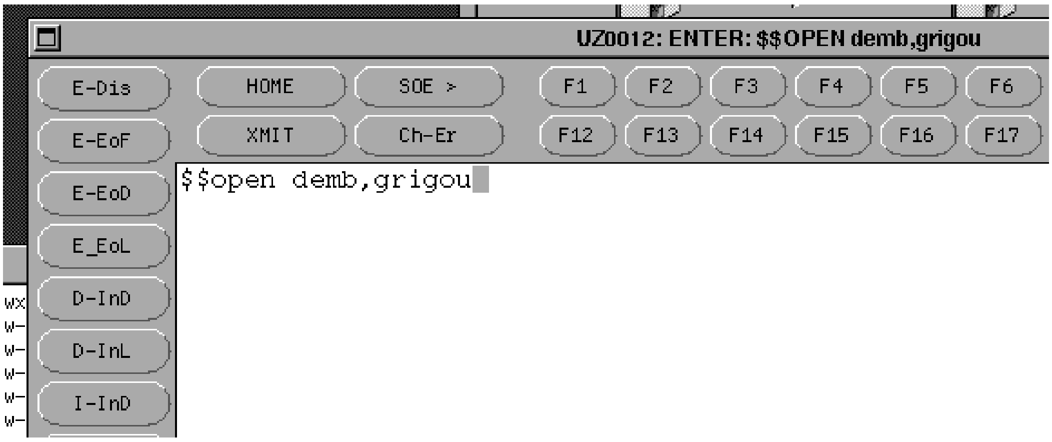
REMARK: Default comes from the following configuration file entries:

Print_Status_Line *0*

6.2.5 Buttons Show, Buttons Hide and Buttons Toggle

`Buttons Show`, `Buttons Hide` and `Buttons Toggle` allows you to switch between the window with buttons and without buttons.

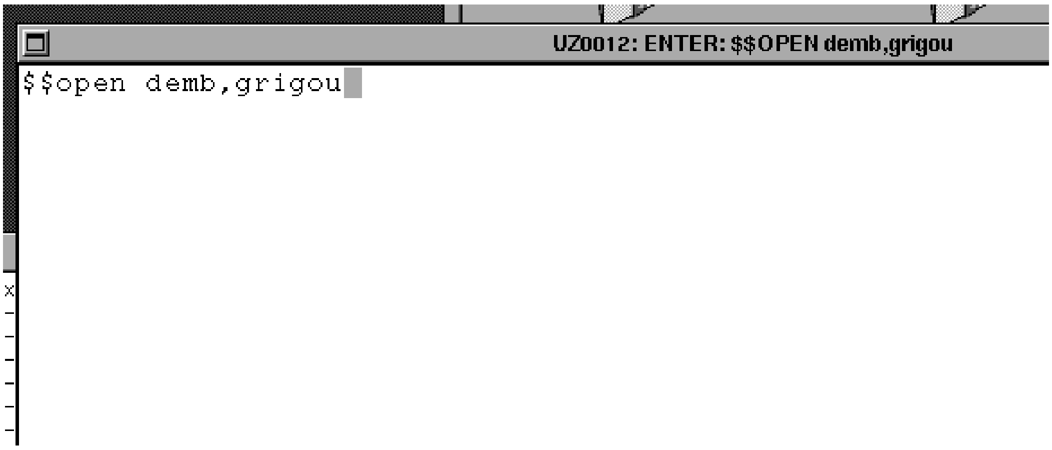
1. Example of window with buttons



UZ0012: ENTER: $$OPEN demb,grigou

```
$$open demb,grigou
```


2. Example of window without buttons

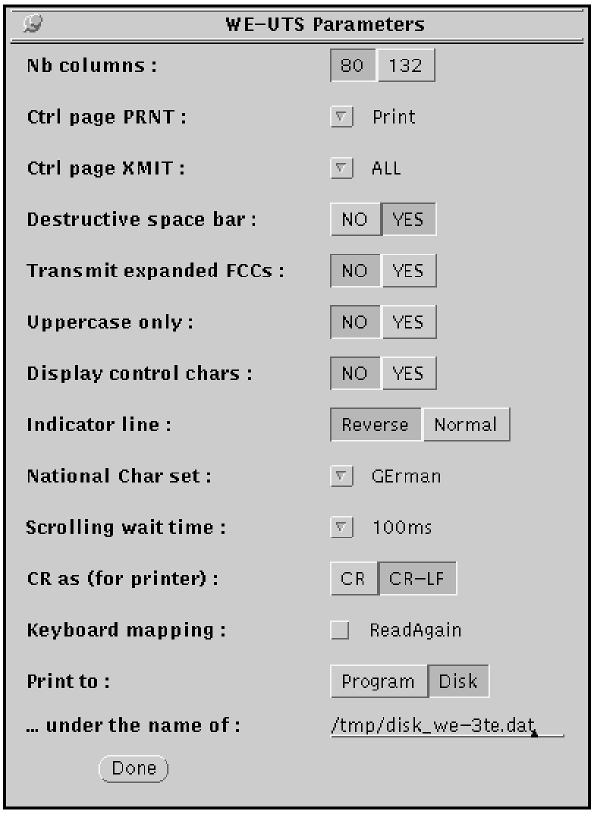


UZ0012: ENTER: $$OPEN demb,grigou

```
$$open demb,grigou
```

6.2.6 Parameters

If you choose the parameter entry, the following window appears:



WE-UTS Parameters

Nb columns : 80 132

Ctrl page PRNT : Print

Ctrl page XMIT : ALL

Destructive space bar : NO YES

Transmit expanded FCCs : NO YES

Uppercase only : NO YES

Display control chars : NO YES

Indicator line : Reverse Normal

National Char set : GErman

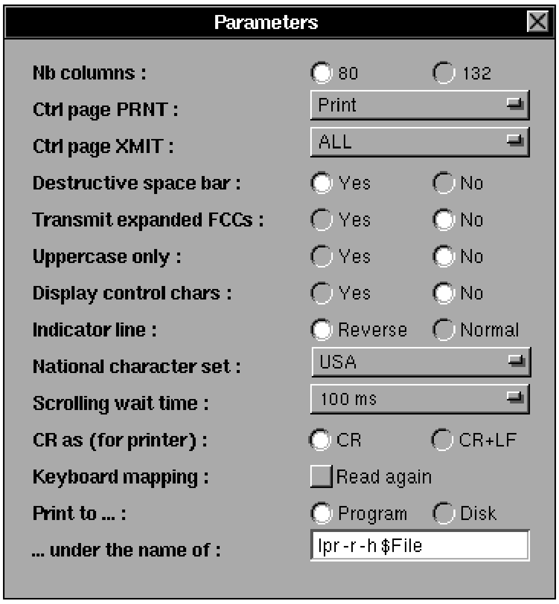
Scrolling wait time : 100ms

CR as (for printer) : CR CR-LF

Keyboard mapping : □ ReadAgain

Print to : Program Disk

... under the name of : /tmp/disk_we-3te.dat



Parameters

Nb columns : ○ 80 ○ 132

Ctrl page PRNT : Print

Ctrl page XMIT : ALL

Destructive space bar : ○ Yes ○ No

Transmit expanded FCCs : ○ Yes ○ No

Uppercase only : ○ Yes ○ No

Display control chars : ○ Yes ○ No

Indicator line : ○ Reverse ○ Normal

National character set : USA

Scrolling wait time : 100 ms

CR as (for printer) : ○ CR ○ CR+LF

Keyboard mapping : □ Read again

Print to ... : ○ Program ○ Disk

... under the name of : lpr -r -h $File

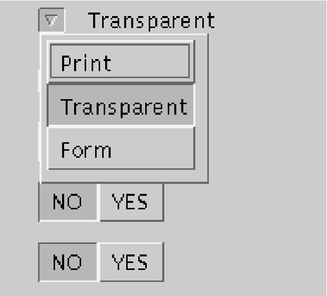
In the parameter window you can set all parameters described below.

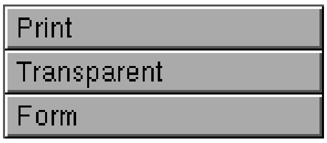
Nb columns

This field allows you to dynamically alter the number of characters per line. (The same you can do in the control page of an SVT terminal.) You may choose between 80 and 132 characters per line. The default corresponds either to 80 or to the <Screen_Cols> entry of your parameter file.

Ctrl page PRNT

This field defines the format of the data transmitted to the printer when the print key is pressed. It is accessible **both** to the operator and the host. The entries are as follows:





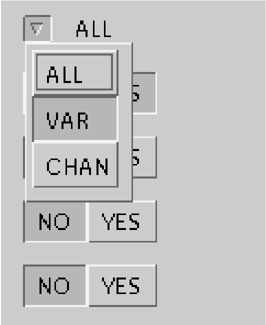
-> Print (default): Trailing spaces on each line are suppressed (except in the line containing the cursor), and the carriage return (CR) code is added at the end of each line.

-> Transparent: Causes a print transparent operation, all CR codes are suppressed, no FCC's are included, and no spaces are suppressed.

-> Form: Trailing spaces on each line are suppressed (except in the line containing the cursor), and spaces are printed instead of protected characters. The CR code is added at the end of each line.

Ctrl page XMIT

This field defines the format of the data transmitted to the host when the XMIT key is depressed. It is accessible **both** to the operator and the host. The entries are as follows:



-> ALL (default): When you depress XMIT, all fields are xmitted to the host.

-> VAR: When you depress XMIT, only the variable fields are xmitted to the host.

-> CHAN: When you depress XMIT, only the modified fields are xmitted to the host.

Destructive space bar

-> YES (default): Pressing the space bar over existing character will replace them with spaces.

-> NO: The space bar will have the same function as the right arrow. It will only step to the next position without replacing characters with spaces.

Transmit expanded FCC's

-> NO (default): WE-UTS will convert expanded FCC's to standard FCC's before xmitting them to the host.

-> YES: WE-UTS will xmit expanded FCC's to the host.

REMARK: In either case, expanded FCC's coming from the host will be accepted and handled correctly.

Uppercase only

-> NO (default): WE-UTS will display both upper and lower case characters.

-> YES: WE-UTS will convert lower to upper case before display.

Display control chars

-> NO (default): WE-UTS will execute (not display) control characters (i.e.: FCC's) coming from the host.

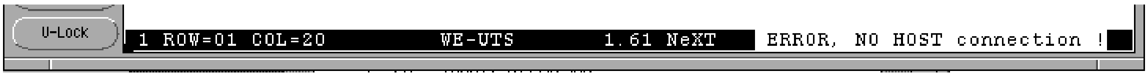
-> YES: WE-UTS will display (not execute) control characters coming from the host.

REMARK: Set this option to YES for debugging purposes only (i.e.: To find out which kind of FCC's or control characters come from the host).

Indicator line

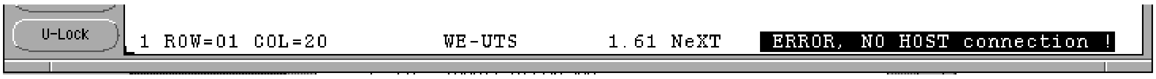
With the indicator line parameter you can set indicator line reverse or normal.

-> Reverse (default): WE-UTS status line (also called indicator line) will be displayed in reverse video.



```
1 ROW=01 COL=20          WE-UTS          1.61 NeXT  ERROR, NO HOST connection !
```

-> Normal: WE-UTS status line will be displayed in normal video.

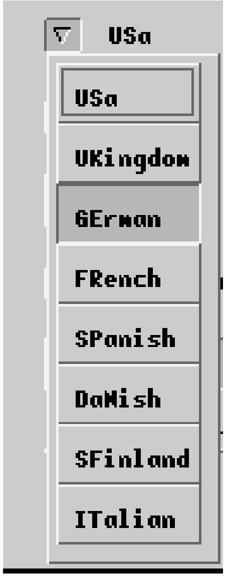


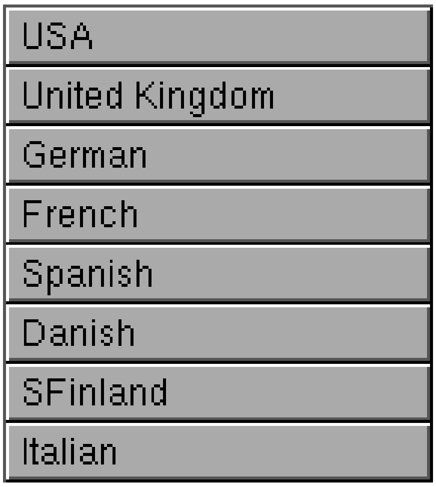
```
1 ROW=01 COL=20          WE-UTS        1.61 NeXT    ERROR, NO HOST connection !
```

REMARK: The concept of reverse and normal video means that the “Foreground” and “Background” colors applied to the terminal area are inverted for the status line.

National character set

With the pull down menu you can choose the national character set.





-> US (default): The Domestic (USA) character set will be used.

-> UK: The United Kingdom character set will be used.

-> GE: The German character set will be used.

-> FR: The French character set will be used.

-> SP: The Spanish character set will be used.

-> DN: The Danish/Norwegian character set will be used.

-> SF: The Swedish/Finnish character set will be used.

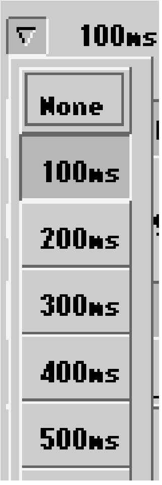
-> IT: The Italian character set will be used.

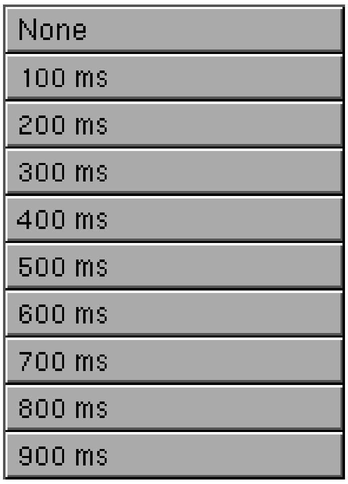
These character sets are as follows:

Standard character sets	**Hex**	**23**	**24**	**40**	**5b**	**5c**	**5d**	**5e**	**60**	**7b**	**7c**	**7d**	**7e**
	Dec	**35**	**36**	**64**	**91**	**92**	**93**	**94**	**96**	**123**	**124**	**125**	**126**
Domestic (USA)		#	$	@	[	\	]	^	‘	{	|	}	~
Spain		Pts	$	§	i	Ñ	¿	^		°	ñ		~
Denmark/Norway		#	$	@	Æ	Ø	Å	^	‘	æ	ø	å	—
France		£	$	à	°	ç	§	^	‘	é	ù	è	¨
Germany		#	$	§	Ä	Ö	Ü	^	‘	ä	ö	ü	ß
Sweden/Finland		#	¤	É	Ä	Ö	Å	Ü	é	ä	ö	å	ü
United Kingdom		£	$	@	[	\	]	^	‘	{	|	}	~
Italy		£	$	§	°	#	é	^	ù	à	ò	è	ì

The remaining characters always correspond to the standard ASCII (7 bits) character set.

Scrolling wait time





When you are working in scroll (demand) mode, it may be necessary to regulate the speed at which data scrolls in your WE-UTS window. This entry allows you to set the minimum time between line scrolls from 0 (no limitation) to 9 (900 milliseconds). You can modify this entry at any time during your terminal session.

CR as (for printer)

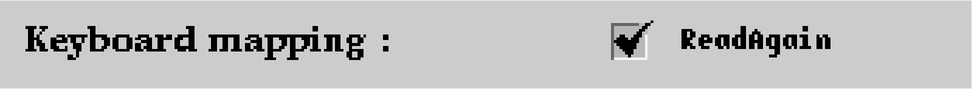
Defines which character is sent to the printer for a line feed.

-> CR (default): When sent to the printer, the CR (Carriage Return) character will not be modified.

-> LF: The CR will be replaced by LF (Line Feed) before being sent to the printer.

Keyboard mapping

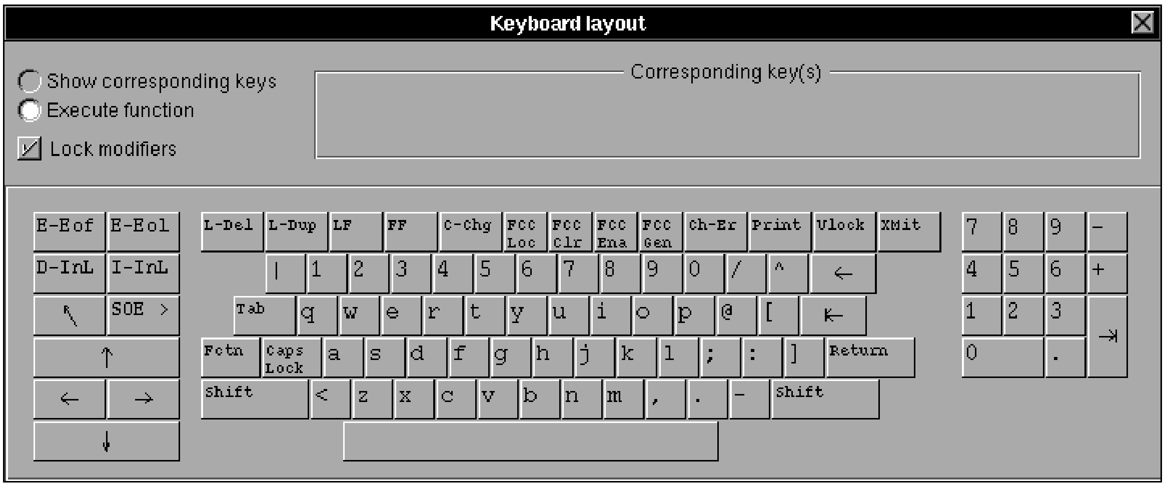
This entry allows you to reread the keyboard mapping file without leaving the WE-UTS emulation. Both “Keymap_File” and “Macro_File” are reread when you click ReadAgain. This is mostly used at installation time to test a newly created keyboard mapping file. Information about the keyboard mapper are found in chapter 7 on page 51.



Keyboard mapping : ☑ ReadAgain

6.2.7 Keyboard layout

If you choose the keyboard layout entry, the following window appears:



Keyboard layout

○ Show corresponding keys
○ Execute function
☑ Lock modifiers

Corresponding key(s)

This is a picture representing an original UTS keyboard. Like on the original, you may enter data, move the cursor, transmit data and so on. Because you have only one mouse (but ten fingers) and some functions or characters may only be produced by depressing several keys simultaneously, we introduced the

Lock modifiers

Button, which allows you to lock the special keys (will highlight when locked) and change the meaning of some keyboard areas accordingly. This is the same principle as (Shift Lock) on a typewriter. To unlock a modifier key, simply click on the highlighted key a second time.

Execute function mode

When this mode is active, any key clicked with your mouse will enter a character into the emulator window or execute some function (i.e.: XMIT, Tab, F6, etc.). Beside your workstation’s physical keyboard and the buttons around the emulator window, this is a third way to enter data into your Unisys emulation window.

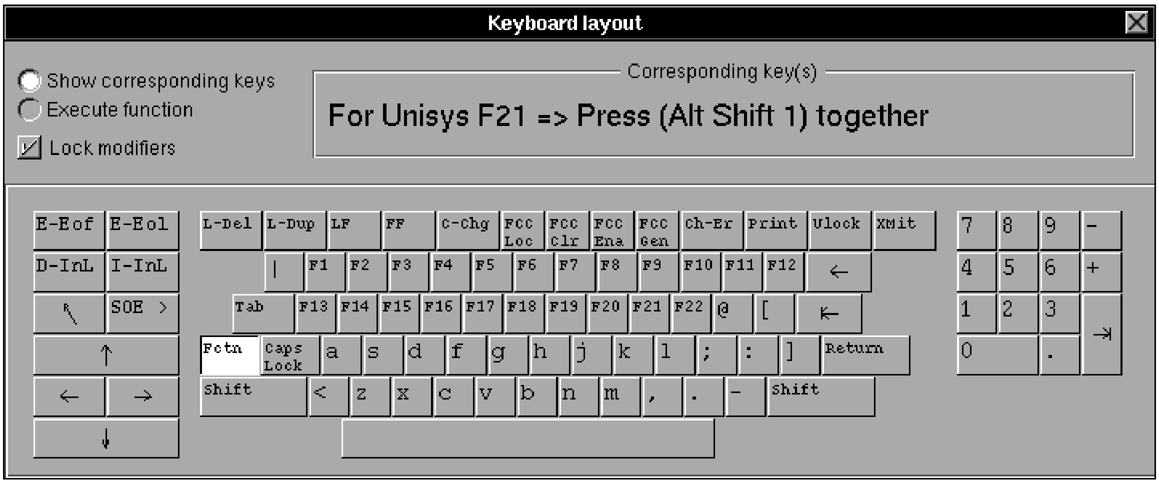
Show corresponding keys mode

When this mode is active, any key clicked with your mouse will display the actual key combination you should type on the physical keyboard of your workstation to obtain the same function as on the original UTS keyboard displayed here.

The assignment of a key, a combination or a sequence of keys to a particular UTS function is done with the ***keymapper tool*** described in chapter 7 on page 51. The show mode described here has been created to help you remembering the assignments you have done.

REMARK: When you click on the keyboard layout in show mode, either the ***key combination*** or the ***comment*** you have entered for this particular combination is shown (see picture bolow). Because the key combination can be somewhat cryptic (machine understandable only!), we warmly recommend that you enter comments which actually describe the key combination in an understandable way.

If a comment is available for an entry, it will always be displayed by the emulator.



Keyboard layout

○ Show corresponding keys
○ Execute function
☑ Lock modifiers

Corresponding key(s)

For Unisys F21 => Press (Alt Shift 1) together

This picture shows you a typical show mode output. The user pressed Fctn which remained highlighted (active) because lock modifiers was set. Then, he clicked on F21 and the following line was displayed:

For Unisys F21 => Press (Alt Shift 1) together

=> This is the comment field which was entered with the ***keymapper tool***.

NOTE 1: In fact, when the user presses “Alt Shift 1” together, some code (i.e.: “Shift Meta /exclamdown”) will be sent from your window server to the WE-UTS emulation program which, in turn, will send F21 to your Unisys mainframe.

NOTE 2: The keyboard mapping can’t be modified from within the emulation. If you want to modify your key assignments, change or add comments, we recommend the following procedure:

How to modify your keyboard mapping

1. Look in your configuration file for the name of the keyboard mapping file (the <Keymap_File> entry). For our example, we will assume that its name is <km.dat>.

2. Start an emulation with this configuration file and connect to the Unisys mainframe.

3. Start a keymapper tool (KM-UTS) with the same keymap file (km.dat) as parameter.

4. Make any changes you want using the “keymapper tool”. When you are done, don’t exit the keymapper tool but use the ***Save*** button to save your changes to the keymap file.

5. Go back into the emulation, display the parameter (Goodies) panel and click the ***ReadAgain*** button. This will read your just saved <km.dat> into the current emulation session and make it immediately active.

6. Try your new key combinations within the emulation. If you are happy with your changes, you may now terminate the “keymapper” process. Otherwise, you may repeat steps 4 to 6 as many times as you want.

Common problems and errors

– When you enter comments in the “keymapper tool”, don’t forget to press the return key to terminate it. Otherwise, the comment will be discarded.

– In step 4, don’t forget to save before proceeding to step 5 and activating ReadAgain. Otherwise, you will read the old version of your keymapping and **no** change will occur.

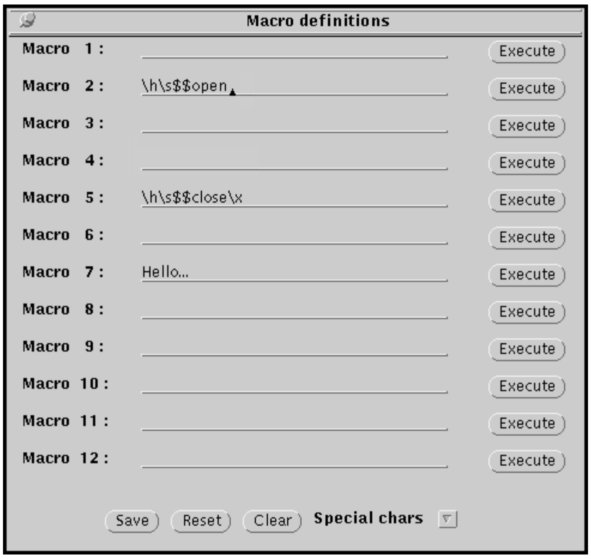
– In step 4, you should comment all entries with a sentence stating clearly what keys are to be depressed to produce the corresponding UTS function.

– If you don’t know anymore which key combination you have assigned to a UTS function, display the “keyboard layout” (Goodies), and use it in show mode.

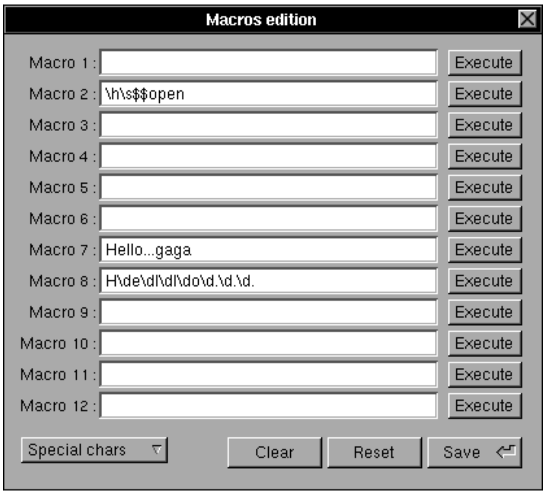
NOTE: For more informations about keyboard mapping, please refer to chapter 7 on page 51.

6.2.8 Macro definitions

If you choose the macro definitions entry, the following window appears:



Macro 1 :
Macro 2 : \h\s$$open
Macro 3 :
Macro 4 :
Macro 5 : \h\s$$close\x
Macro 6 :
Macro 7 : Hello...
Macro 8 :
Macro 9 :
Macro 10 :
Macro 11 :
Macro 12 :



Macro 1 :
Macro 2 : \h\s$$open
Macro 3 :
Macro 4 :
Macro 5 :
Macro 6 :
Macro 7 : Hello...gaga
Macro 8 : H\de\dl\dl\do\d.\d.\d.
Macro 9 :
Macro 10 :
Macro 11 :
Macro 12 :

Purpose of this panel is to allow you to examine, execute or modify your current macro (key sequences) assignments. The actual content of the entries has been read from the <Macro_File> entry of your current configuration.

Clear button

When clicked, this will clear **all** macro entries. If you clicked it by mistake, you still can retrieve your macro entries by clicking the Reset button (see below).

Reset button

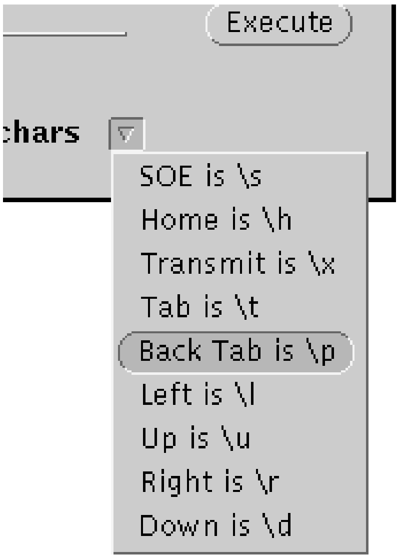
When clicked, this will revert **all** macro entries to the last saved version.

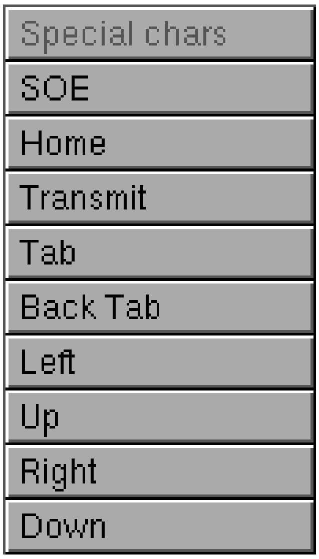
Save button

When clicked, this will save your current entries to disk. The file in which these macro entries will be saved is indicated by the <Macro_File> entry of your current configuration.

Special chars

When clicked, the following menu will be displayed:



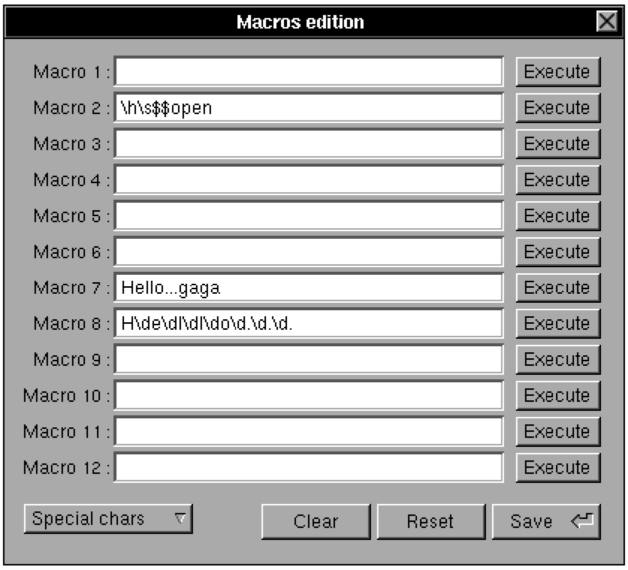


Here is how to proceed

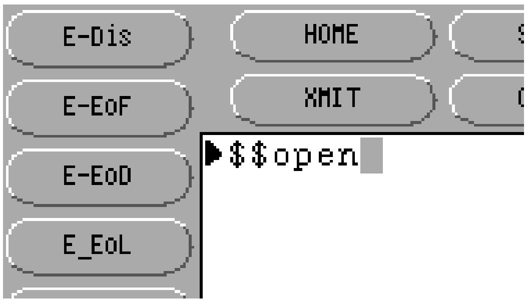
Let’s say you want that macro 2 (when activated) puts the cursor to home and displays an SOE followed by $$OPEN.

1. Click at the beginning of the macro 2 field with the mouse.
2. For NEXTSTEP:
   With the mouse, click on special chars and select the “Home” entry. When selected, release the mouse button. This will automatically enter a “\h” entry at the beginning of the macro 2 field.

   For X windows:
   With the mouse, click on special chars and read the indication on the right of the “Home is” entry. Then, enter the value read (\h) into the macro 2 field.
3. Repeat step 2 for the SOE entry.
4. Enter $$OPEN in the macro 2 entry after the “\h\s”.
5. Try this macro by clicking on the Execute button sitting on the right of the macro 2 entry. Look at the pictures on page 46 for the result of these actions in the macro panel and in the WE-UTS emulation window.



Macro 1 :
Macro 2 : \h\s$$open
Macro 3 :
Macro 4 :
Macro 5 :
Macro 6 :
Macro 7 : Hello...gaga
Macro 8 : H\de\dl\dl\do\d.\d.\d.
Macro 9 :
Macro 10 :
Macro 11 :
Macro 12 :




```
▶$$open
```

REMARK: 12 keys (or key combinations) may be assigned for the execution of macros without displaying the present panel. For that purpose, you must use the “Keymapper tool” on page 51 and enter key combinations for the M_MACRO1EXEC to M_MACRO12EXEC entries.

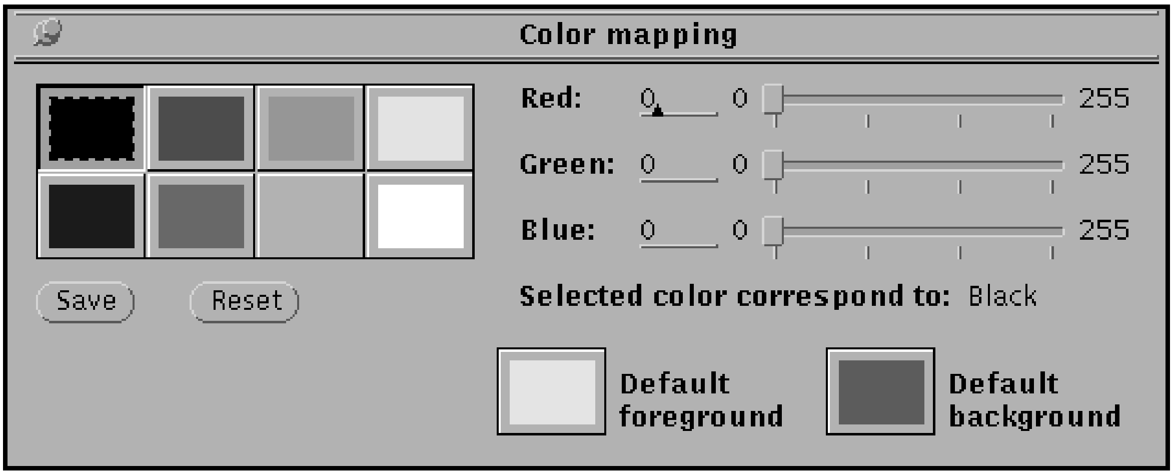
6.2.9 Color mapping

Purpose of the color mapping utility is to allow the user to change the original UTS colors to your preferred colors. The UTS protocol specifies 8 colors for text (WE-UTSc) and 16 colors for graphic (WE-UTSg).

At emulation start-up, the color mapping comes from the <Color_File> entry of the configuration file. This file will be modified when you click the Save (X) or OK (NEXTSTEP) button of the color mapping panel.

6.2.10 Color mapping under X

When you select the color mapping entry of the Goodies menu, one of the windows on page 47 appears (depending if you have graphic capabilities or not).



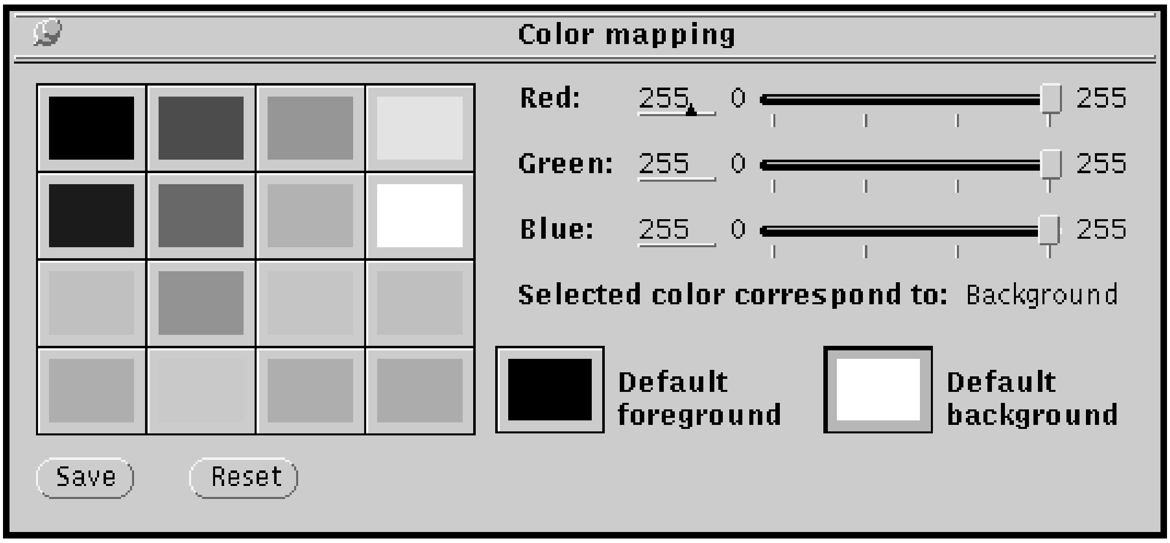
Red: 0 0 255

Green: 0 0 255

Blue: 0 0 255

Selected color correspond to: Black

X window systems: valid for WE-UTSc (character mode terminal uses 8 colors)



Red: 255 0 255

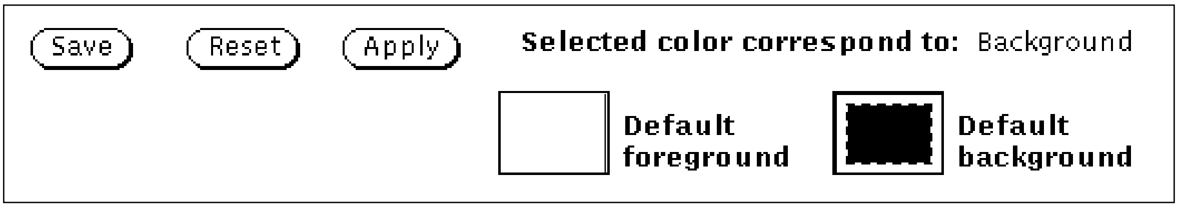
Green: 255 0 255

Blue: 255 0 255

Selected color correspond to: Background

X window systems: valid for WE-UTSg (graphic terminal uses 16 colors)

If your X server has a Static Color Map, the lower part of the color mapping panel (see the two pictures on page 47) will look as below:



WE-UTS will switch automatically between Dynamic and Static Color Maps. If supported by your X server, Dynamic Color Maps will be used.

Reset button

When you click this button, the colors are reset to the values of the last saved version.

Save button

This button allow you to save your changes to disk. This action can't be undone.

How to proceed

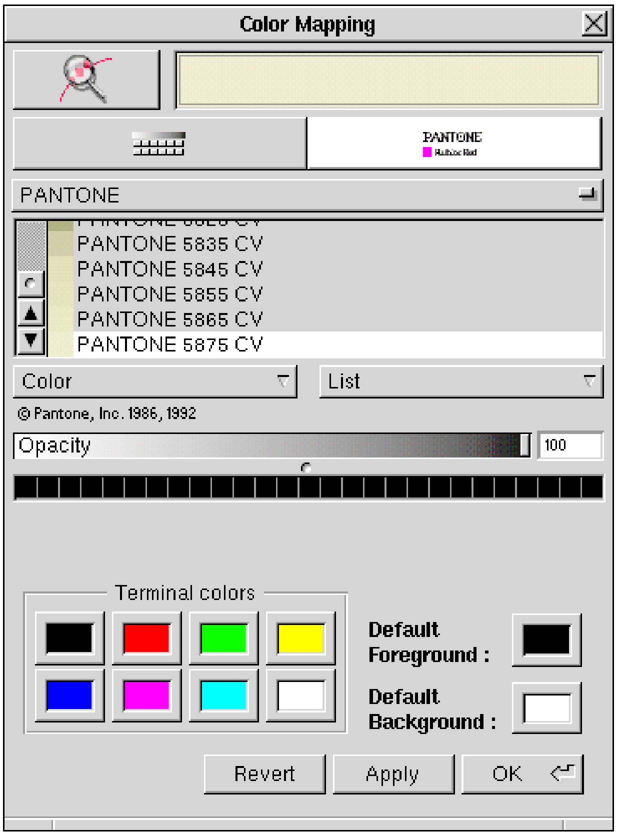
1. Connect to the mainframe and display a colorful screen.
2. In the Goodies menu, select color mapping. One of the color editing tool panel (see pictures on page 47) will appear.
3. Within the panel, click on the color you want to modify. The name of the color you clicked on will appear in the panel (i.e.: background).
4. Use the three cursors to modify the color:

   -> For *Dynamic Color Maps*, the result will be immediately visible in the color mapping tool and in your emulator character window while moving the slider.

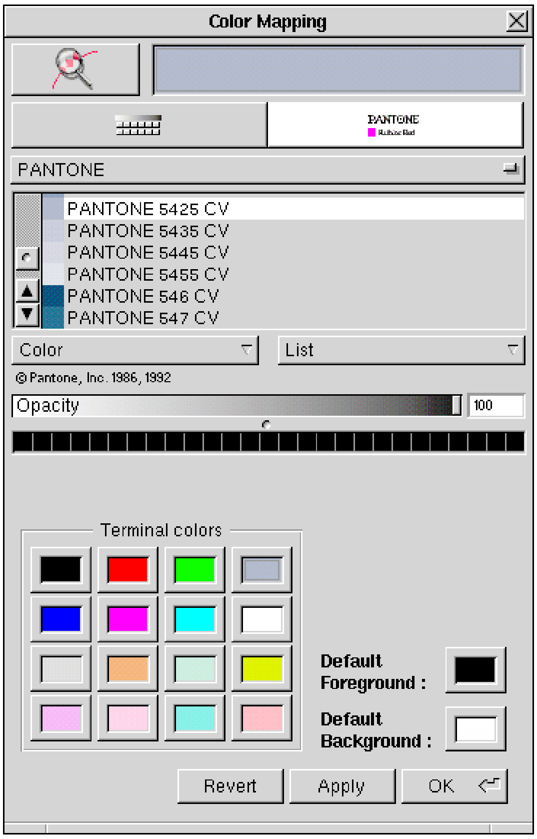
   -> For *Static Color Maps*, the result will be visible only in the color mapping tool after you have released the slider. To make the change appear in the emulator window, click the supplementary **Apply button**.
5. When you are done, click the Save button to save your work in the file indicated with <Color_File> in the active configuration (i.e.: color.dat).

6.2.11 Color mapping under NEXTSTEP

When you select the color mapping entry of the Goodies menu, one of the window on page 49 appears (depending if you purchased the WE-UTSc <character only> or the WE-UTSg <graphic> terminal emulation).



NEXTSTEP without graphic (WE-UTSc)



NEXTSTEP with graphic (WE-UTSg)

REMARK: WE-UTS uses a NeXT object for the color edition. This object may be improved from one NEXTSTEP release to another and WE-UTS will benefit of those enhancements. Therefore, we will not describe the object itself, but only its relations with our particular application. For more informations about the NeXT color object and how to use it, please refer to your NeXT “User’s Reference Manual”.

Apply button

Click on this button to make your color changes appear in your emulator character (graphic) window without leaving the color mapping tool.

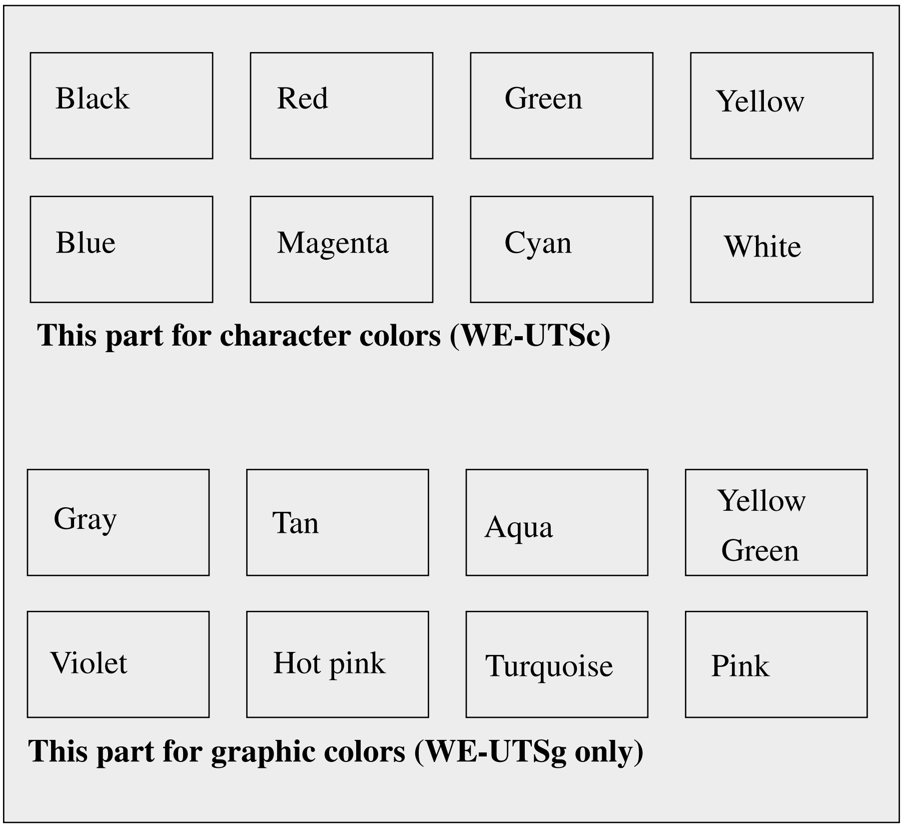
Revert button

Click on this button to revert the colors to what they where when you started the emulator.

OK button

Click on this button to exit the color mapping tool. Your changes will be saved to disk and applied to your emulator window (as with the Apply button).

Correspondence between UTS colors and mapping tool colors



Black	Red	Green	Yellow
Blue	Magenta	Cyan	White

This part for character colors (WE-UTSc)

Gray	Tan	Aqua	Yellow Green
Violet	Hot pink	Turquoise	Pink

This part for graphic colors (WE-UTSg only)

How to proceed

1. Connect to the mainframe and display a colorful screen.

2. In the Goodies menu, select color mapping. One of the color editing tool panel shown above will appear (the panel aspect may change depending on your NEXTSTEP release).

3. Create your custom color the same manner you would do in any other NEXTSTEP application. When you are done, drag this color into one of the 8 (16) UTS colors at the bottom of the panel or into the “Default Foreground” or “Default Background” color. The color will change to the one you dragged in. Click the Apply button to reflect this change in your emulator window. Repeat this step until you are satisfied with all colors.

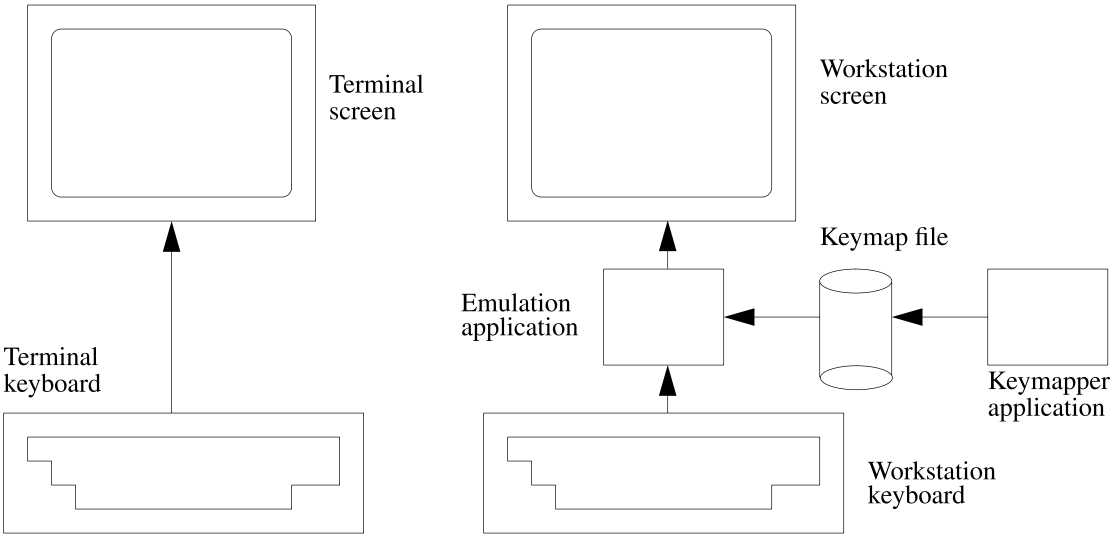
4. When you are done, click the OK button to save your work in the file indicated with <Color_File> in the active configuration (i.e.: color.dat). The color mapper tool panel will be closed.

5. If you want to reset the colors to what they were at emulation start-up time, you may click the Revert button. All your changes will be discarded.

7 Keymapper tool

7.1 Introduction

The keymapper is a tool application to perform keyboard mapping operations. The keyboard mapping is to assign some or all keys of the original terminal keyboard to your workstation keyboard. It is particularly useful to enter original terminal “keys” that have no correspondence on your workstation keyboard.



Keyboard mapping isn’t integrated in the terminal emulation application because it is intended to be used by the system manager. Communication between the keymapper and the terminal emulation application is done through a file, the keyboard mapping file. This file contains associations between keys or key combinations and terminal keys or functions. The main purpose of the keymapper application is to provide an easy way for interactively building or editing the keyboard mapping file.

7.2 Definitions

Before describing further how to work with the keymapper application, some definitions must be stated:

7.2.1 A terminal function

is a function provided by the original terminal (and by the terminal emulation). Terminal functions can be (but aren’t necessarily) obtainable via the original terminal keyboard. Example of terminal functions (keys are enclosed in quotes): “Return”, “Backspace”, “Next Field”.

7.2.2 Terminal keys

are alphabetical, numeric and other ASCII symbol keys.

7.2.3 A modifier key

is a key like shift or ctrl. Its purpose is to change the code associated to a set of keys. Modifiers known by the application are shift, ctrl, meta (alternate on some keyboards) and numeric lock (num-lock). It is possible that you cannot access all of these modifiers with your workstation keyboard.

7.2.4 A key combination

is a single key typed with or without simultaneously stroking a modifier key. Example of key combinations (keys are enclosed in quotes): “a”, “A”, “Ctrl”-“c”, “Shift”-“Meta”-“F1”. You can notice that “A” is not, but could be “Shift”-“a”: every workstation company has its own policy for assigning codes to its keyboard(s) keys, that is why a keyboard mapping tool is useful.

7.2.5 An escape sequence

are keys typed one after the other (without modifiers) proceeded by a dedicated key combination (can have modifiers) called the **escape sequence introducer key combination** (or more quickly the **sequence intro key**). The purpose of escape sequences is to increase the possibilities of function mapping when the workstation keyboard offers too few possible key combinations. Example of escape sequences (the escape intro key is “Meta”-“e”, keys are enclosed in quotes, sequences are separated by commas): “Meta”-“e” “f” “c”, “Meta”-“e” “F6”, “Meta”-“e” “t” “r” “y”.

NOTE: For VT320 emulation users:
Escape sequences can replace the “Compose” key mechanism. You can assign all compose keys to escape sequences.

7.2.6 A special key

is a key that doesn’t generate an alphanumerical character. Special keys are keys used to do formatting (carriage return, tabulations, moving cursor), function keys or diacritical signed characters (é, ˆ, Ç, etc.). Special keys have names, always starting with a slash (‘/’); they are listed in chapter 7.7 on page 58.

7.2.7 An exotic key

is a key that doesn’t have a corresponding name. An exotic key is handled with an immediate (cryptic) value instead of a name. Exotic keys should be rare on standard keyboards.

7.3 Installing and starting the keymapper tool

7.3.1 Foreword

This chapter is common for the whole WE-XXX product family. Presently, this includes the

WE-UTSc / WE-UTSg / WE-D320 / WE-I3179c / WE-I3179g

products. Since all pictures were done with the *WE-UTSc* product, the illustrations shown may be somewhat different for other products. However, all buttons, functions and procedures are common to all products.

7.3.2 Under NEXTSTEP

Copy the application into the "LocalApps" or "NeXTApps" folder, or in the "Apps" folder below your home folder. Copy the keyboard mapping file anywhere you want. Copy the keyboard layout file into your home folder (if you want to start the application by a double-click on its icon) or in the current directory (if you want to start the application through a shell window). The keyboard layout file must be named "keylay.dat" or the "-klf" command line option should be used (for more see "Tips for advanced users" on page 57).

To be recognized as the document of the keymapper application (visualized with the same icon than the application), the keymapping file must have the "*xxx*_keymap" extension (where *xxx* is the name of the terminal emulation <UTS, D320, I3179>). Like every NeXT application, the application must reside in one of the standards application folders.

7.3.3 Under X windows

Copy all files from the distribution media into a directory, and type in a shell window:

```
cd program_directory
km-xxx km.dat
```

NOTE: Depending on the particular WE products, "km.xxx" can be one of the following:

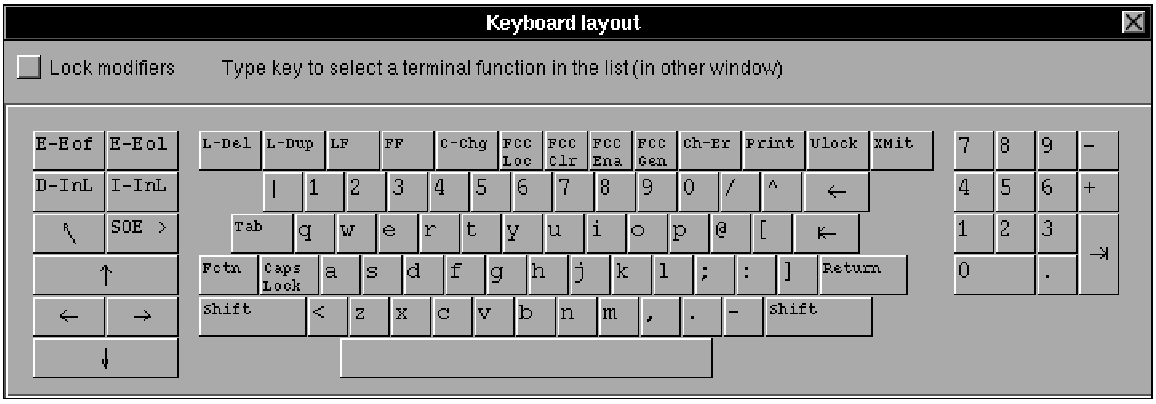
km-uts	for the WE-UTS emulation
km-d320	for the WE-D320 emulation
km-i3179	for the WE-I3179 emulation

Then proceed with the next chapter.

7.4 Using the keymapper tool

7.4.1 Usage overview

To start the keymapper application, type its name in a command shell or double-click on the keymapping file icon. After a few seconds, two windows should be visible: the first containing an image of the original terminal keyboard, the other containing a list of terminal functions and some buttons, this is the keymapper main window.



Keyboard layout

□ Lock modifiers

Type key to select a terminal function in the list (in other window)

To make associations between original keyboard functions and key combinations of your physical workstation keyboard, proceed as follows:

1. Click any key of the keyboard layout window: you can see that the corresponding terminal function (or key) is selected.

2. Now select the main window and type any key or key combination on your workstation physical keyboard. You can see that the "name" of this key is displayed at the left of the terminal function list: you have created an association between the selected terminal function and the displayed key combination.

NOTE 1: In step 1, you could also directly select a terminal function by choosing an item of the terminal function list.

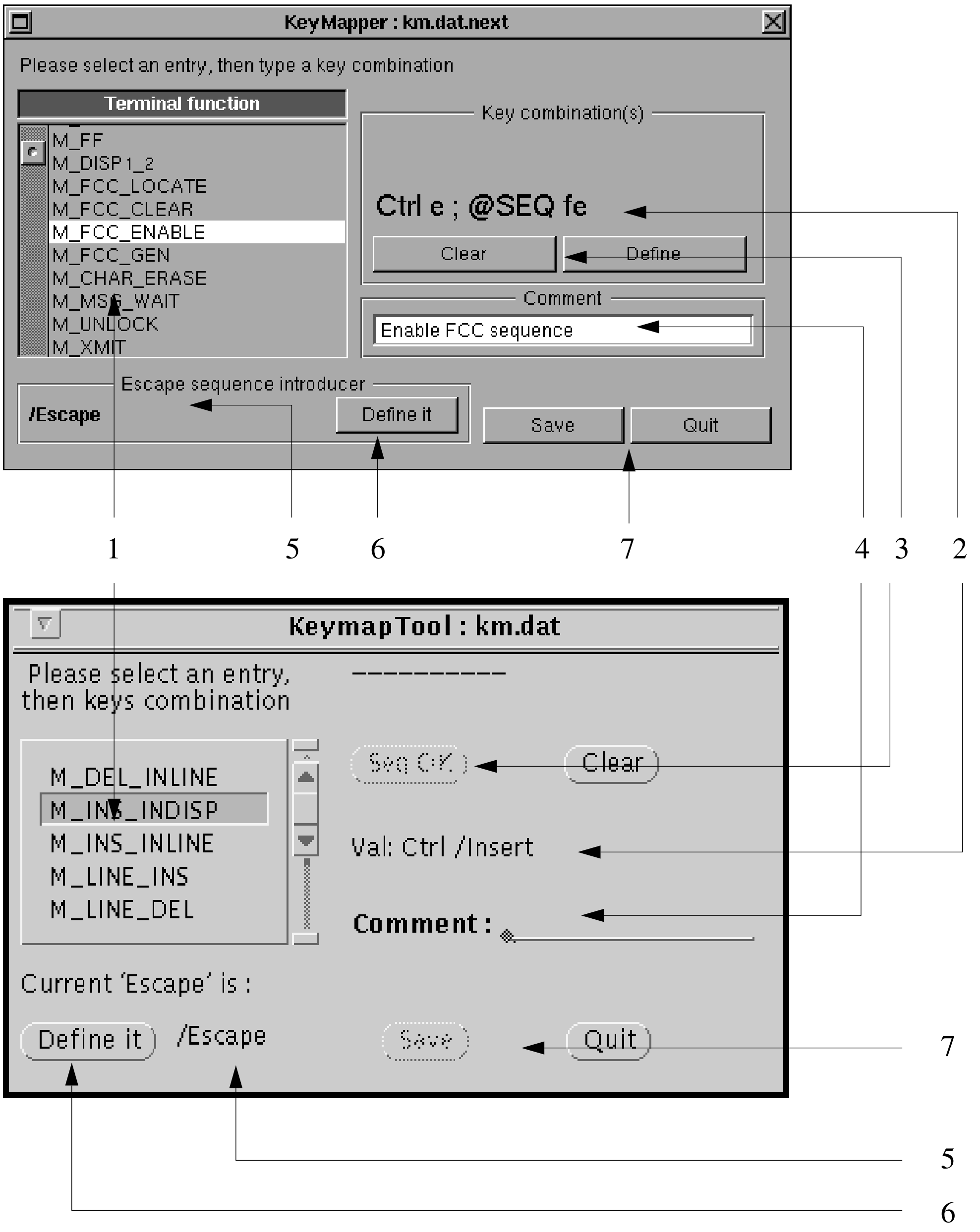
NOTE 2: (For NeXT users only): You cannot use the "Command" modifier in the key combinations you define, because "Command" should be used only for menu shortcuts (accelerators), and terminal functions are not provided in the application menus.

The illustrations on page 55 show the main window of the keymapper tool.

(1) The list of the terminal functions and keys. All terminal functions are preceded by "M_" to avoid confusion with keys.

(2) The key combination associated to the currently selected terminal function (or key).

(3) The Define and Clear current key combination buttons.



KeyMapper : km.dat.next

Please select an entry, then type a key combination

Terminal function

M_FF
M_DISP1_2
M_FCC_LOCATE
M_FCC_CLEAR
M_FCC_ENABLE
M_FCC_GEN
M_CHAR_ERASE
M_MSG_WAIT
M_UNLOCK
M_XMIT

Key combination(s): Ctrl e ; @SEQ fe

Comment: Enable FCC sequence

Escape sequence introducer: /Escape

KeymapTool : km.dat

Please select an entry, then keys combination

M_DEL_INLINE
M_INS_INDISP
M_INS_INLINE
M_LINE_INS
M_LINE_DEL

Val: Ctrl /Insert

Comment :

Current 'Escape' is : /Escape

(4) The comment associated to the currently selected terminal function (or key). You can edit this comment.

(5) The current escape sequence introducer key.

(6) The escape sequence introducer key definition button. After clicking this button you can define a new escape sequence introducer key by typing a key combination.

(7) The Save and Quit buttons.

7.4.2 Terminal function list

You can select a terminal function by clicking on an item of the list and browse through the list by clicking on the arrow buttons. You can notice that when selecting an item, the key combination field and comment change accordingly.

The list begins with the terminal functions (preceded by the "M_" sequence), continues with all the regular ASCII characters and terminates with the extended international characters.

NOTE: By default (in the delivered "km.dat" file), only the regular ASCII characters are mapped to their equivalent.

7.4.3 Key combination field

A terminal function or key can be associated with one or several key combination or escape sequences. If there are several key combinations (or escape sequences) they are separated by semicolon ";". The modifier names are displayed in extension, the key name is either a letter or a string, for special keys (see chapter 7.7 on page 58). The sequence intro key is indicated with the "@SEQ" string. Exotic keys are introduced with the "Hexa" string followed by a number in hexadecimal base.

The application denies associating a key combination to two different terminal functions or keys. It shows a warning panel on the screen and cancels the entered key combination. You must remove the key combination associated to the other terminal function to associate it to the currently selected terminal function. Similarly you cannot define an escape sequence that is already included in another one: for example, the "@SEQ fe" cannot be defined if the "@SEQ fec" already exists.

7.4.4 Notes for NeXT users

– There is no difference between the numeric keypad and the numeric keys on the main keypad, because the NEXTSTEP environment returns the same codes for both key sets.

– You may get some strange results when typing key combinations. For example, if you type the "2" key and simultaneously press the "Ctrl" modifier, "Ctrl" will be displayed in the key combination field.

In fact, NEXTSTEP does part of the job of the keymapper application: when using key combination with modifiers (shift, alternate, ctrl) it returns the corresponding code. For example, when typing "Ctrl"-"A", NEXTSTEP doesn't return "A" with modifier "Ctrl" but the code of the control-A character (in our case ASCII code 01) with "Ctrl" modifier.

This behaviour can be handled by the keymapper application for "Ctrl" modifier on alphabetical keys, but with "Ctrl" modifier on other keys, and especially for "Alternate" modifier there is no way to retrieve the key combination from the code returned by NEXTSTEP (e.g. there is no way to retrieve the typed "Meta-s" from the received "Meta-ß" ["Meta /ssharp"] because on another NeXT keyboard [on a keyboard for a different country, for example] the "Meta-s" can generate another code).

7.4.5 Define and Clear buttons

These buttons allow you to change the key combination associated with a terminal function or key. Hit the “Clear” button to wipe-out all the previously defined key combinations of the currently selected terminal function. Check the “Define” (“SEQ end”) button to notify the end of the escape sequence currently being defined.

7.4.6 Comment field

To edit a comment, click in the comment field, type your new comment and don’t forget to **press the return key** to indicate that you’ve finished editing a comment (otherwise you won’t be able to enter new key combinations; they will accumulate in the comment field).

NOTE: We warmly recommend that you enter comments which actually describe the key combination (or sequence) you have pressed to produce the codes. For example, if you pressed “Alternate-t” to be assigned to, let’s say home, a possible comment would be:

For home press <Alt and t> simultaneously.

The comment field gets displayed in the <keyboard layout> window of the emulation when set in <show mode>. The user may read it by clicking on a key in the <keyboard layout> window (Goodies menu) while in show mode.

7.4.7 Escape sequence intro key

To edit or set the escape sequence introducer key you must click the “Define” button and then type the key combination you want. This will change the sequence intro key for all future and already defined escape sequences.

NOTE: You may define only one sequence intro key.

7.4.8 Save and Quit buttons

The “Save” button allows you to save the current keyboard mapping definitions. The “Quit” button is for quitting the application: a panel is prompted, providing a last chance to save the modifications you made to the definitions, if any.

7.5 Tips for advanced users

7.5.1 Command line parameters

There are some options that you can add to the command line when starting the application. They allow you to customize the behaviour of the application.

-nc *national_charset_name*

Forces the application to display the given national variant of the original terminal keyboard. This affects the keyboard layout window. The different national character set names are defined in the emulation documentation, for the same option (“-nc”). The default national variant used is the US one.

-lnr *number_of_entries*

Specifies the number of entries that can be simultaneously displayed in the terminal function list (affect the size [in pixels] of the scrolling list). Setting the number too high (typically > 20) will have an unpredictable effect.

-klf *name_of_keyboard_layout_file*

Specifies the name and the path of the keyboard layout file, the file describing the layout of the terminal's original keyboard. The default name of the keyboard's layout file is "keylay.dat". The default path is the current directory.

-flk *name of font*

Allows to change the font used for the labels of the keyboard layout keys. The font name must honour the syntax used by display environment (X or NEXTSTEP). Choosing an unappropriate font may cause problems (partially hidden key labels). Default font used is courier. For X, this font must be available at least in 10, 12 and 14 points size.

7.6 Editing the keyboard mapping file

There is a possibility to manually edit the keyboard mapping file. However, you will loose some of the checks made in the keymapper application and there is no guarantee that the key combinations you define can be generated by your workstation keyboard.

The syntax used in the keymapping file is described in the file itself.

7.7 Names of the special keys

Return	Return key
/Backspace	Delete previous character
/Tab	Forward tabulation
/RevTab	Backward tabulation (NeXT only)
/Up, /Left, /Right, /Down	Arrow keys (for moving the cursor)
/Escape	Escape key
/KP_*x*	Numeric keypad keys (*x* is a digit or other symbol)
/F*x*	Function keys
/*x*acute	(*x* is a vowel) á, é, í, ó, ú
/*X*acute	(*X* is a capital vowel) Á, É, Í, Ó, Ú
/*x*grave	(*x* is a vowel) à, è, ì, ò, ù
/*X*grave	(*X* is a capital vowel) À, È, Ì, Ò, Ù

/*x*diaeresis	(*x* is a vowel) ä, ë, ï, ö , ü, ÿ
/*X*diaeresis	(*X* is a capital vowel) Ä, Ë, Ï, Ö, Ü, Ÿ
/*x*circumflex	(*x* is a vowel) â, ê, î, ô, û
/*X*circumflex	(*X* is a capital vowel) Â, Ê, Î, Ô, Û
/aring, /Aring	å, Å
/oslash, /Oslash	ø, Ø
/ntilde, /Ntilde	ñ, Ñ
/otilde, /Otilde	õ, Õ
/ccedilla, /Ccedilla	ç, Ç
/ae, /AE	æ, Æ
/eth, /Eth	
/thorn, /Thorn	
/ssharp	ß
/exclamdown	¡
/cent	¢
/sterling	£
/currency	¤
/yen	¥
/brokenbar	
/section	§
/diaeresis	¨
/copyright	©
/ordfeminine	ª
/guillemotleft	«
/notsign	¬
/hyphen	–
/registered	®
/macron	¯

/degree	°
/plusminus	±
/onesuperior	
/twosuperior	
/threesuperior	
/acute	´
/mu	µ
/paragraph	¶
/periodcentered	·
/cedilla	¸
/masculine	º
/guillemotright	»
/onequarter	
/onehalf	
/threequarters	
/questiondown	¿
/multiply	×
/divide	÷

A WE-UTS parameter file format

```
# WE-UTS WINDOW PARAMETERS

Screen_Rows          24                     # Number of rows on your screen is 24
Screen_Cols          80                     # Default value for the column number is 80
Point_Mouse          1                      # Cursor positioning with the mouse enabled
Buttons_Panel        1                      # Buttons will be displayed around the window
Exit_Confirm         1                      # Will bring exit panel at exit time
Scroll_Delay         1                      # Allow 100 ms between lines
Lock_Setup           0                      # 1st priority configuration file will NOT be locked

# WE-UTS BLINKING OPTIONS

Blink_Cursor         1                      # The cursor will blink
Blink_OutFocus       1                      # Blinking fields will continue to blink for ever
Inactiv_Time         15                     # Stop blinking anyway after 15 seconds

# WE-UTS CHARACTER SET OPTIONS

National_Cset        US                     # The default character set

# WE-UTS CHARACTER FONTS (used by WE-UTSc and WE-UTSg)

Font_R               -b&h-*-medium-r-*-*-%d-*-*-*-m-*-iso8859-1
Font_B               -b&h-*-bold-r-*-*-%d-*-*-*-m-*-iso8859-1
Font_Dimensions      8, 10, 12, 14, 18, 19, 24   # The sizes for the font window
Start_Height         12                     # Will start with font size = 12
Font_P               -b&h-*-medium-r-*-*-%d-*-*-*-m-*-iso8859-1
Font_Warning         1                      # Will print warnings for font problems
Invert_Bold          1                      # Low intensity will be displayed with bold font

# WE-UTS GRAPHIC FONTS (used by WE-UTSg only)

Graphics_Font   courb.pfa                   # Graphic font used by UTS 60 emulation
Graphics_Width 800                          # Start width of a graphical window
Graphics_Height 800                         # Start height of a graphical window

# WE-UTS SERVER AND HOST CONNECTION OPTIONS

Term_Name            UZ0012                 # The terminal PID as defined in CMS
Term_Type            7                      # UTS 40 type (for DCP/TELNET)        NOTE 1
Appl_Name            demb                   # Default application name
# Appl_Server        TIPCSU                 # CSU name (for HLC/Tp0)              NOTE 1
Comd_Host            localhost              # The host on which <comd> is running
Comd_Service         we-comd                # Service between <comd> and <we-uts>
Mainframe_Host       MyUnisysHost           # DCP name on Internet network
Mainframe_Service    UnisysDcp              # Service between <DCP> and <comd>

# NOTE 1:  Either <Term_Type> or <Appl_Server> may be present in a configuration file, not both.
# WE-UTS AUXILIARY FILES

Keymap_File         km.dat                  # User definable keymapping
Macro_File          macro.dat               # User definable macros
Kbd_Layout_File     keylay.dat              # Layout of the original UTS keyboard
Color_File          color.dat               # User definable colors
Buttons_File        button.dat              # Your custom buttons definitions

# WE-UTS PRINT PARAMETERS

Disk_Print          ~/we-uts.$ext           # Destination for disk print
Print_Program       lpr -r -h $File         # Destination for program print
Print_Dest          Disk                    # Destination is DISK
Print_Status_Line   1                       # Add status line and border
Print_Format        Postscript              # Format is PostScript
Print_Output_Size   A4                      # (PostScript only) Paper size
Print_Orientation   Landscape               # (PostScript only) Orientation
Print_Scale         80                      # (PostScript only) Scaling
Print_Generic_File  ~/we-uts/ps_generic.dat # (PostScript on X only) Skeleton

# WE-UTS PASSWORD

Pass_Word           PutYourPasswordHere     # The WAG supplied password
```

We will now discuss each parameters in detail and specify whether they are mandatory, optional or if they have default values.

NOTE: All entries must be in the parameter file format (see page 17).

A.1 Parameter description

The entries marked with an * can be modified at run time by clicking in the WE-UTS parameter panel.

WE-UTS WINDOW PARAMETERS

1. Screen_Rows and Screen_Cols

– Definition: These two parameters specify how many active rows and columns your WE-UTS window will have.

– Defaults: 24 rows and 80 cols

– Mandatory: No

NOTE: A typical value of 80 columns and 24 rows is required by most Unisys applications. Some applications written for the SVT may require 132 columns.

2. Point_Mouse

– Definition: If set to 1 (YES), the left mouse button in the WE-UTS window will place the cursor on the position pointed or to the next unprotected screen position.

– Defaults: 0 (NO)

– Mandatory: No

3. Buttons_Panel *

– Definition: If set to 1 (YES), clickable buttons representing the special keys of an WE-UTS terminal (i.e.: function keys) will be displayed around the WE-UTS terminal window. You will then be able to activate all these special functions by clicking them with the mouse.

– Defaults: 0 (NO)

– Mandatory: No

NOTE: At any time during the WE-UTS session you can toggle this option using the WE-UTS window menu. This option defines only the start-up value for this parameter.

4. Exit_Confirm

– Definition: If you set this entry to 1 (YES), you will get a panel when you choose the <exit> or <quit> entry in the application menu. This will give you a chance to continue your session.

– Defaults: 0 (NO)

– Mandatory: No

5. Scroll_Delay *

– Definition: This entry defines the minimum time between line scrolls (demand mode). You can preset values between 1–9 (100–900 milliseconds). Setting this option to 0 will scroll at the maximum possible speed.

– Defaults: 0 (Highest scroll speed: NO delay)

– Mandatory: No

6. Lock_Setup

– Definition: If set to 1 (YES), the primary setup file (see “Hints about locking configuration file(s)” on page 27) will be locked (reserved) for the whole duration of the WE-UTS session.

You may use this option only if you have write access to the primary setup file.

– Defaults: 0 (NO)

– Mandatory: No

WE-UTS BLINKING OPTIONS

7. Blink_Cursor

– Definition: If set to 1 (YES), the cursor will blink. Otherwise, it will be a solid black rectangle. Note that the cursor may stop to blink when the WE-UTS window is out of focus (see “Blink_OutFocus” below).

– Defaults: 0 (NO)

– Mandatory: No

8. Blink_OutFocus

– Definition: If set to 1 (YES), blinking fields will continue to blink in a non active window (out of focus). If set to 0 (NO), they will stop to blink thus lowering the processor load for that inactive window.

– Defaults: 1 (YES)

– Mandatory: No

9. Inactiv_Time

– Definition: This option gives the number of seconds after which any blink activity (cursor, blinking fields, etc.) will stop in case of inactivity. This will be effective even if the window is “In Focus” (is the active window). Values may range from 1 to 255 seconds. 0 means infinite time.

– Defaults: 0 (Infinite)

– Mandatory: No

WE-UTS CHARACTER SET OPTION

10. National_Cset *

– Definition: This is the name of the default character set that will be used for your WE-UTS session. Refer to the Unisys host system administrator for the correct character set to use at your particular site.

– Defaults: US

– Mandatory: No

WE-UTS CHARACTER FONTS

11. Font_R and Font_B

– Definition: These entries define the fonts to be used for the text (data) of your WE-UTS session. The `Font_R` is used for normal intensity display and the `Font_B` is used for highlight (bold) display. Both fonts **must** exist and **must** be accessible to your window server. For more informations see “Fonts specifications” on page 76.

– Defaults: None

– Mandatory: Yes

12. Font_Dimensions

– Definition: Here you must place all sizes of `Font_R` and `Font_B` that you want to use in the WE-UTS session. Note that all the sizes listed have to be available to your server. The sizes can be separated by spaces or commas.

– Defaults: None

– Mandatory: Yes for X windows / No for NEXTSTEP

13. Start_Height

– Definition: Here you have to put the size of `Font_R` and `Font_B` that you want to use at WE-UTS start-up time. It must be one of the sizes listed above.

– Defaults: None

– Mandatory: Yes

14. Font_P

– Definition: This entry defines the font to be used for the buttons in the panels (see “Buttons_Panel *” on page 63). The font given here **must** exist and **must** be accessible to your window server. For more informations see “Fonts specifications” on page 76.

– Defaults: None

– Mandatory: Yes

NOTE: The font size chosen here will define the minimum size of the WE-UTS window while the buttons are displayed.

15. Font_Warning

– Definition: If you set this entry to 1 (YES), you will get more detailed messages, if the server has some trouble while loading the fonts you specified. This may be useful if you are suspecting a font loading problem.

– Defaults: 0 (NO)

– Mandatory: No

16. Invert_Bold

– Definition: An original Unisys UTS 30 can display normal and low intensity characters. If you set this entry to 0 (NO), then the `Font_R` (regular) will be used to display low intensity characters, while the `Font_B` (bold) will be used for normal intensity. This complies with to the original but may not be the best solution in every situation.

If you set this entry to 1 (YES), the regular font will be used to display normal intensity characters. The bold font will be used for the low intensity characters thus mostly enhancing the readability of the display.

– Defaults: 0 (NO)

– Mandatory: No

WE-UTS GRAPHIC FONTS

17. Graphics_Font
– Used by WE-UTSg graphic emulation only –

– Definition: The name of the scalable font used for a UTS 60 graphic window.

Under X, you can use any of the three public domain fonts provided by workstation ag:

-> cour.pfa (standard courier font)
-> courb.pfa (courier font bold)
-> couri.pfa (courier font italic)

Under NEXTSTEP, you can use any monospaced font like:

-> Courier (standard courier)
-> Ohlfs

– Defaults: None

– Mandatory: Yes, if you want to display graphics.

18. Graphics_Width

– Definition: The horizontal start-up size (in pixels) of an UTS 60 graphic window.

– Defaults: None

– Mandatory: Yes, if you want to display graphics.

19. Graphics_Height

– Definition: The vertical virtual start-up size (in pixels) of an UTS 60 graphic window. We call it <virtual>, because a window created only 73 % of its height. This is necessary to get a 1/1 aspect ratio and therefore get <round> circles.

REMARK: As a rule, choose always the same number for the <Width> and the <Height> parameter. Doing so, you will automatically get a 1/1 aspect ratio.

– Defaults: None

– Mandatory: Yes, if you want to display graphics.

WE-UTS SERVER AND HOST CONNECTION OPTIONS

20. Term_Name

– Definition: The name given here will be used as a terminal ID for the Unisys host connection. Therefore, it **must** exist and be declared in the Unisys host configuration (i.e.: CMS 1100 PID statement). If not, the Unisys host will refuse the connection and you will get an error message. Note that the same name can’t be used for more than one connection at any given time. This restriction is imposed by Unisys.

– Defaults: None

– Mandatory: Yes

21. Term_Type
- For TELNET connections to DCP Telcon™ only -

– Definition: The type given here will be used while establishing the connection with the Unisys host (DCP). Possible entries are 7 (UTS 40 single station) or 12 (UTS 60). It is very important that you set this entry to the correct value. Otherwise, color or graphic mode may not be possible. For more informations see “About colors and graphics” on page 79.

– Defaults: None

– Mandatory: Yes

22. Appl_Name

– Definition: This is the name of the application to which WE-UTS will try the first connection. It will also appear on the window border and on the WE-UTS icon while the connection is established and active.

– Defaults: None

– Mandatory: No

NOTE: The “Appl_Name” can also be given (and changed) using the $$OPEN and $$CLOSE orders inside the WE-UTS window.

23. Appl_Server
- For Tp0 connections to HLC only -

– Definition: The parameter given here will be sent to the Unisys host (HLC) while establishing the connection. It is also known as the TSAP (Transport Service Access Point) or TSU (Transport Service User). Possible values are:

TIPCSU (for TIP applications)
RSDCSU (for DEMAND applications)

– Defaults: None

– Mandatory: Yes

NOTE: The “Appl_Server” can also be given (and changed) using the $$OPEN and $$CLOSE orders inside the WE-UTS window.

CAUTION: A valid configuration can contain **either** *<Term_Type>* **or** *<Appl_Server>* but not both. *<Term_Type>* is used on TELNET (DCP) connections while *<Appl_Name>* is used on Tp0 (HLC) connections.

24. Comd_Host

– Definition: This is the name of the (UNIX) host on which the protocol server of the emulation is running. Since the “WE-COMD” component of the emulation may be loaded and running separately on each workstation of the network running WE-UTS or be common for all, it may be necessary to specify on which (UNIX) host it is running.

– Defaults: The host on which WE-UTS is running.

– Mandatory: No, if protocol server is running on the same (UNIX) host as WE-UTS.

NOTE 1: If the protocol server is not running on the same (UNIX) host as WE-UTS, then this entry is mandatory. Otherwise, the protocol server will not be accessible and you will get an error message at WE-UTS start-up time.

NOTE 2: If the machine, on which WE-UTS is running, is part of a YP (NIS) network, remember that “localhost” is no longer the host on which you are working, but the YP server.

25. Comd_Service

– Definition: This is the name of the service under which the “WE-COMD” component is waiting for the WE-UTS connection requests. The name you give here **must** match the one given while starting the “WE-COMD” component and must have been declared in “/etc/services”, YP (Yellow Pages = NIS) or Netinfo (NeXT).

– Defaults: None

– Mandatory: Yes

26. Mainframe_Host

– Definition: This is the name of the (UNISYS™) host to which WE-UTS will try the first connection. It will also appear on the window border while the connection is established and active.

– Defaults: None

– Mandatory: No

NOTE: The “Mainframe_Host” name can also be given (and changed) using the $$OPEN and $$CLOSE orders typed inside the WE-UTS window.

27. Mainframe_Service

– Definition: This is the name of the service under which the “WE-COMD” component of the WE-UTS emulation will try to connect to the (UNISYS™) host. This must have been declared in “/etc/services”, YP (Yellow Pages = NIS) or Netinfo (NeXT).

– Defaults: None

– Mandatory: Yes

WE-UTS AUXILIARY FILES

28. Keymap_File

– Definition: The name of the file containing the definitions for your keyboard mapping. You may modify this file using the “KM-UTS” utility. Please refer to the “Keymapper tool” on page 51 for a complete description of this utility.

– Defaults: None

– Mandatory: Yes, if you want to map function keys, use international character or macros.

NOTE: We deliver a file called “km.dat” with the emulation. This is a template that you may modify to match your requirements.

29. Macro_File

– Definition: The name of the file containing the definitions for up to 12 keyboard macros. Although you may create or modify this file with a common text editor, we recommend that you use the “Macro definition” panel while running the emulation (see “Macro definitions” on page 44).

– Defaults: None

– Mandatory: Yes, if you want to use macros.

NOTE: We deliver a file called “macro.dat” with the emulation. This is a template that you may modify to match your requirements.

30. Kbd_Layout_File

– Definition: The name of the file containing the image of the original Unisys UTS keyboard used by the “KM-UTS” utility. Can also be found in the “Goodies menu” under “keyboard layout”. You can’t modify this file.

– Defaults: None

– Mandatory: Yes, if you want to display the graphical keyboard.

NOTE: This entry should point to the “keylay.dat” file delivered with the application. Only the pathname may be modified.

31. Color_File

– Definition: The name of the file containing the color definitions for your emulation. It is useful, even if you have no color application, because it allows you to choose custom colors for your window background and foreground.

– Defaults: None

– Mandatory: Yes, if you want custom colors.

NOTE: With the emulation, we include a “color.dat” file as a template. You may use it as is or modify it at will.

32. Buttons_File

– Definition: The name of the file containing the buttons definition for your emulation. Format of this file is as follows:

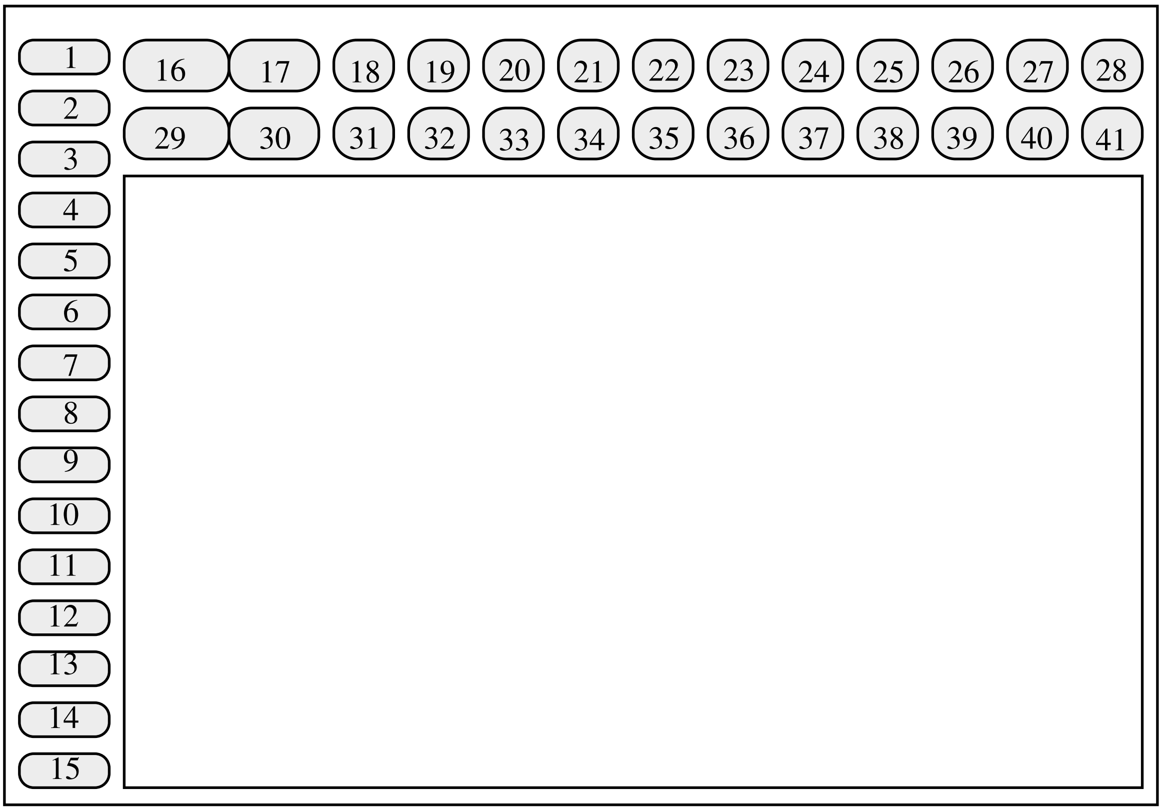
<Index, Button_title, M_func_name>
<Index, Button_title, M_func_name>
<Index, Button_title, M_func_name>
...

Where:

-> “Index”	the fix assigned button number as in figure below.
-> “Button_title”	the text which will appear in the button.
-> “M_func_name”	the emulator function executed when you click this button with your mouse (see Appendix F on page 91).

– Defaults: None

– Mandatory: Yes, if you want custom buttons.
No, if you are happy with the default text and functions of the buttons.



REMARK: -> If you want to have space chars in the Button_title, you must bracket its definition as follows:

<3, “My Button”, M_DO>

Will display My Button in button 3. Clicking this button, will send a DO function code to the attached application.

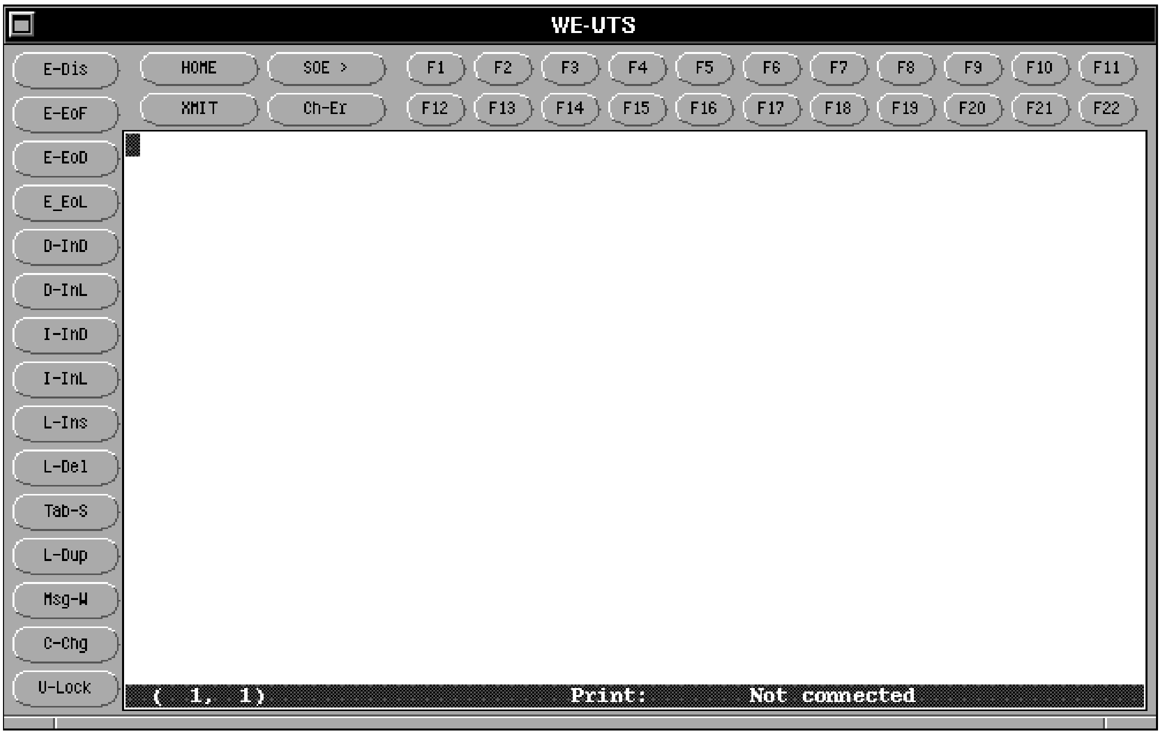
-> On X11 servers, the button width on top-right panel is determined by the length of their title. All other buttons (top-left panel and left panel) have a fixed width.

-> On NeXT, all buttons are of fixed width.

-> Depending on the text length and on the fixed (variable) width of the button, not all text entered may appear in it. Therefore, we encourage you to test your definitions with different window sizes (resize) to help refining them.

-> With the emulation, we deliver a “button.dat” file as a template. You may use it as is or modify it at will.

-> Buttons, for which you give no definitions, have defaults as below:



WE-UTS PRINT PARAMETERS

The different print formats are discussed in chapter 6.2.4 on page 31. Please refer to that chapter for detailed informations regarding printing with WE-UTS.

33. Disk_Print *

– Definition: The name of the file print data will be directed to when the print destination is a disk file.

– Defaults: /tmp/WE-UTS/disk_WE-UTS.dat

– Mandatory: No

– Example: Disk_Print ~/we-uts/MyPrints.$ext

Creates files with sequential numbering in the user’s home directory. See discussion in chapter 6.2.4 on page 31 for the meaning of $ext and how it will be replaced by WE-UTS.

Disk_Print /tmp/MyPrints

Puts the print data in the file “MyPrints” in the /tmp directory. Since (without $ext) this file will be created at each print occurs, it will only contain the data of the **last print** done.

34. Print_Program *

- Definition: The name of the program print data will be directed to when the print destination is a program.
- Defaults: lpr -r -h $FILE
- Mandatory: No

35. Print_Dest *

- Definition: The default destination for your print data. Possible values are:

  Printer
  Disk
- Defaults: Printer
- Mandatory: No

36. Print_Status_Line *

- Definition: Will add the status line and a window border on the printout if set. Values are:

  0 # No status line
  1 # Add status line to printouts
- Defaults: 0 (No status line)
- Mandatory: No

37. Print_Format *

- Definition: Will define in which format printing will occur. Values are:

  Text only
  Formatted
  PostScript
- Defaults: Text only
- Mandatory: No

REMARK: There is currently no difference between Text only and Formatted.

38. Print_Output_Size *

– Definition: Will define the paper size. This entry is only used if "Print_Format" is set to PostScript. Values are:

Letter
A3
A4
A5

– Defaults: Letter

– Mandatory: No

39. Print_Orientation *

– Definition: Will define the paper orientation. This entry is only used if "Print_Format" is set to PostScript. Values are:

Portrait
Landscape

– Defaults: Portrait

– Mandatory: No

40. Print_Scale *

– Definition: Will define the scaling factor in %. This entry is only used if "Print_Format" is set to PostScript. Values are:

from 30 to 300 (decimal value)

– Defaults: 100

– Mandatory: No

41. Print_Generic_File

– Definition: Will define the name and path of the PostScript generic file which WE-UTS will use as a template to generate PostScript:

~/we-uts/ps_generic.dat

– Defaults: None

– Mandatory: Yes, if you want to generate PostScript and you don't work with NEXTSTEP.

REMARK: With the emulation, we deliver a file called “ps_generic.dat”. This is a template that you probably will use as delivered. Experienced PostScript programmers may modify it to suit their special needs.

WE-UTS PASSWORD

42. Pass_Word

– Definition:	A password provided by workstation ag. This must be typed exactly as provided. A password may contain (depending on your particular licence agreement) informations like: – hostid or hostname – expiration date – maximum run time – host(s) allowed to display
– Defaults:	None
– Mandatory:	Yes

NOTE: The password is different for “WE-UTSc” (character only emulation) and “WE-UTSg” (character and graphic emulation). This allows a user to begin with a character emulation and upgrade to the UTS 60 graphics without reinstalling a new product. Only the password will be changed.

A.2 Fonts specifications

1. All fonts used with “WE-UTS” **must** be fixed character width (not proportional) fonts (i.e.: b&h, courier).
2. Both fonts (Font_R and Font_B) must be the same size (width and height).
3. The fonts and sizes you specify must be available to your X server.
4. While the Buttons_Panel is displayed, the window can **never** become smaller than the buttons size. So, if you want to resize your window to a smaller size than the buttons, you **must** first disable them with the “Toggle” menu option.

A.3 Fonts recommended for X window systems

Font_R	-b&h-lucidatypewriter-medium-r-*-*-%d-*-*-*-m-*-iso8859-1
Font_B	-b&h-lucidatypewriter-bold-r-*-*-%d-*-*-*-m-*-iso8859-1
Font_Dimensions	8, 10, 12, 14, 18, 19, 24
Start_Height	12
Font_P	-b&h-lucidatypewriter-medium-r-*-*-%d-*-*-*-m-*-iso8859-1

A.4 Fonts recommended for NEXTSTEP systems

Font_R	Courier
Font_B	Courier-Bold
Font_Dimensions	8, 10, 12, 14, 18, 19, 24
Start_Height	12
Font_P	Ohlfs

B About colors and graphics

As mentioned at the beginning of this manual, the WE-UTS terminal emulation may be purchased in two versions:

WE-UTSc	Emulates nearly all Unisys terminals and includes UTS 60 colors.
WE-UTSg	Same as WE-UTSc plus UTS 60 graphics.

Although the configuration of WE-UTS is quite simple forward, there are some points to remember if you want to display colors and graphics.

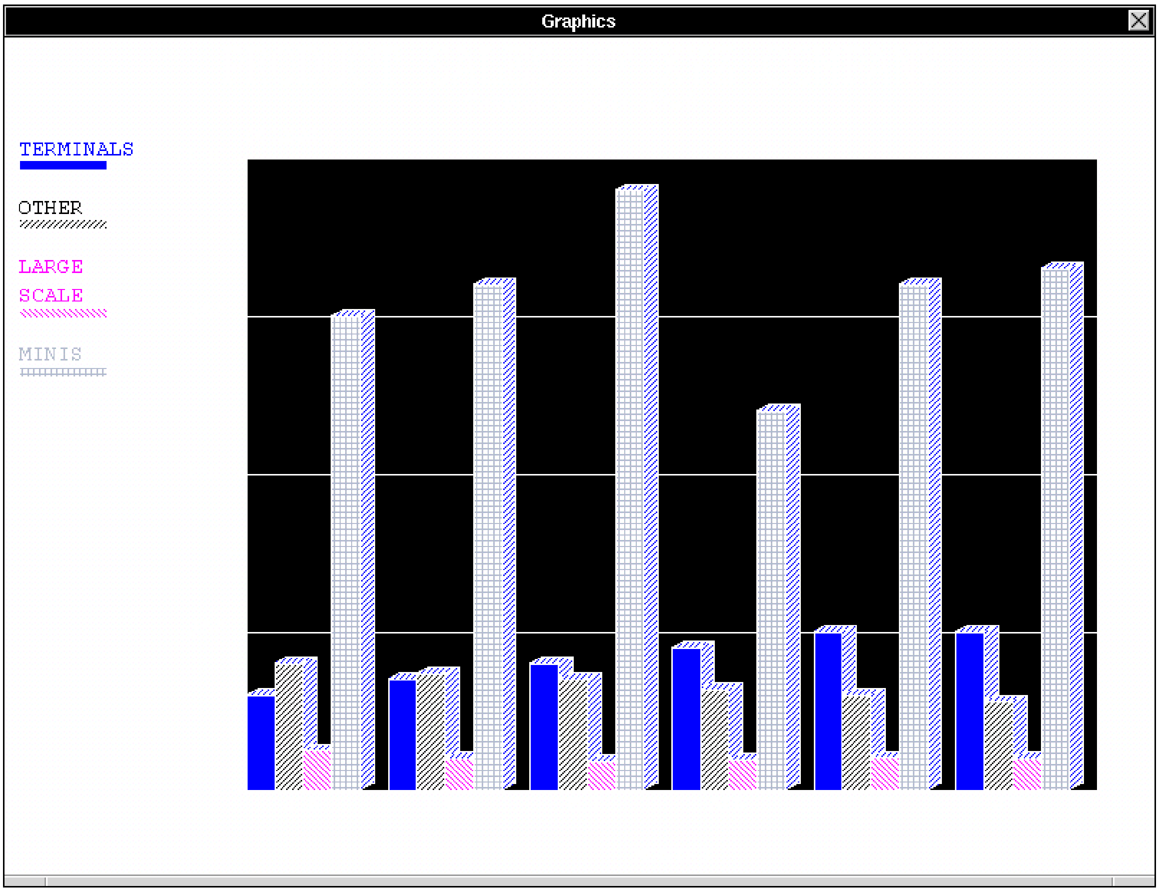
1. If you purchased WE-UTSc, you got a password which disallows UTS 60 graphics. However, the emulation will not crash if graphic orders are received. If this occurs, the graphic orders will merely be displayed as characters in the terminal window instead of being displayed in a separate graphic window.

2. Colors and/or graphics may only be received if your host configurations (CMS, Telcon™) recognizes your terminal ID <Term_Name> parameter entry as being a UTS 60 terminal. Furthermore, don’t forget to set the <Term_Type> parameter entry to 12 if you have a DCP TELNET connection. For more informations about the possible terminal types, refer to the Unisys Telcon™ TCP/IP stack software documentation.

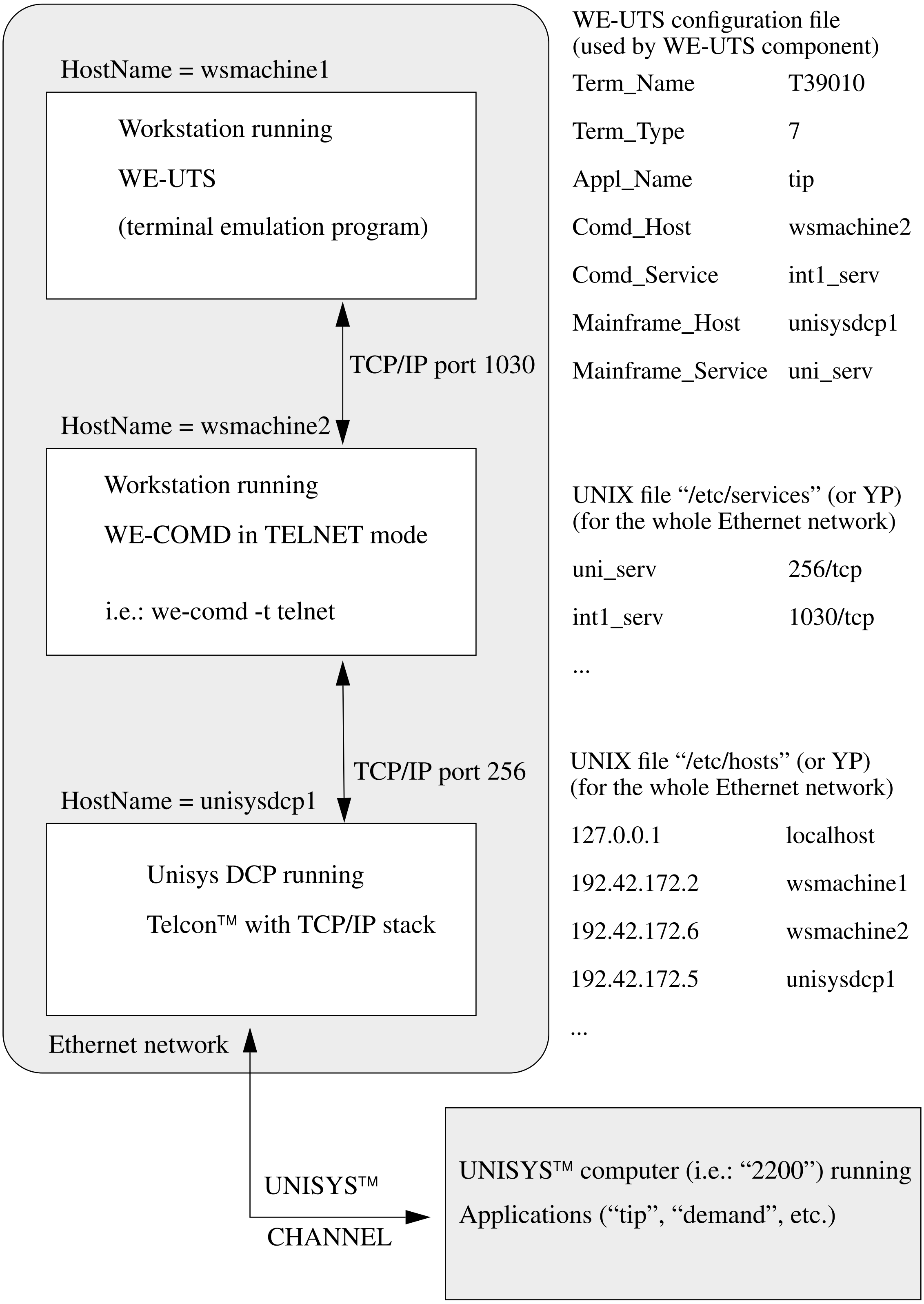
3. When a graphic screen is received, it will **not** be displayed in the same window as the text (as on an original UTS 60 terminal), but in a separate graphic window. This window can be resized, moved, hidden, etc. as any other window in your workspace.

4. The graphic window will always display the last graphic screen received. This means that you can go on with your work within the character window without loosing your last graphical output. When you don’t need this last graphic any longer, simply close this window. If new graphic arrives while the graphic window is closed, a new graphic window will be opened.

The illustration below shows a typical graphic screen output (produced by Unisys mapper).



C DCP TELNET example



WE-UTS configuration file
(used by WE-UTS component)

Term_Name	T39010
Term_Type	7
Appl_Name	tip
Comd_Host	wsmachine2
Comd_Service	int1_serv
Mainframe_Host	unisysdcp1
Mainframe_Service	uni_serv

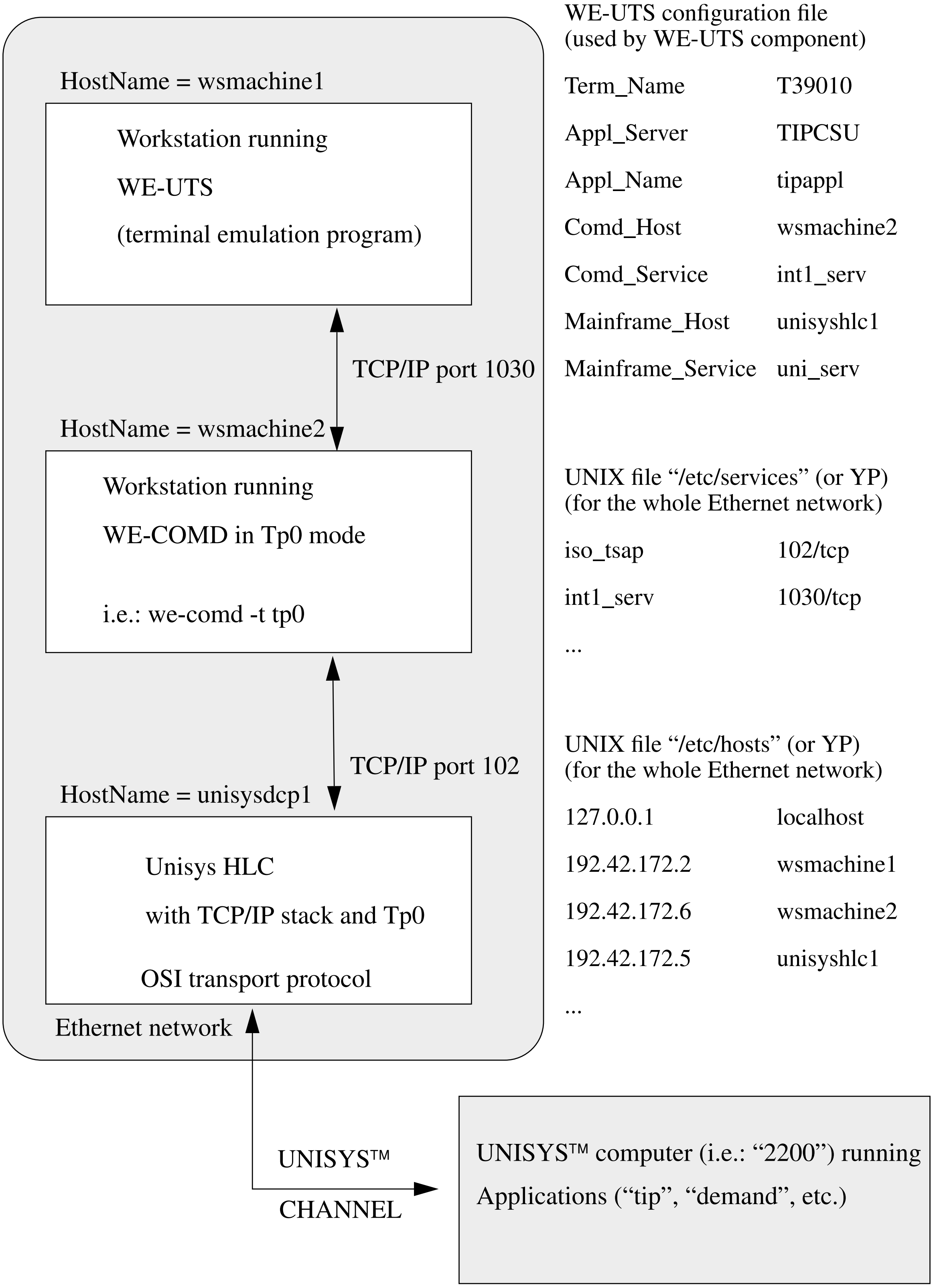
UNIX file "/etc/services" (or YP)
(for the whole Ethernet network)

uni_serv	256/tcp
int1_serv	1030/tcp
...	

UNIX file "/etc/hosts" (or YP)
(for the whole Ethernet network)

127.0.0.1	localhost
192.42.172.2	wsmachine1
192.42.172.6	wsmachine2
192.42.172.5	unisysdcp1
...	

D HLC Tp0 example



WE-UTS configuration file
(used by WE-UTS component)

Term_Name	T39010
Appl_Server	TIPCSU
Appl_Name	tipappl
Comd_Host	wsmachine2
Comd_Service	int1_serv
Mainframe_Host	unisyshlc1
Mainframe_Service	uni_serv

UNIX file "/etc/services" (or YP)
(for the whole Ethernet network)

iso_tsap	102/tcp
int1_serv	1030/tcp
...	

UNIX file "/etc/hosts" (or YP)
(for the whole Ethernet network)

127.0.0.1	localhost
192.42.172.2	wsmachine1
192.42.172.6	wsmachine2
192.42.172.5	unisyshlc1
...	

E Status line, window border line, icon and messages

The status line (25th line of the text window) contains the following fields:

Cursor coordinates	Release field	Communication status field

E.1 Normal messages displayed in the communication status field

E.1.1 Connected to application

This is the normal message which must be displayed when you are connected to any host application. In this case, the upper window border contains a text like:

S39080: Session <demb> on system <wagwest>

Where:

`- S39080`	is your Term_Name (see “Hints about locking configuration file(s)” on page 27)
`- demb`	is the application you are connected with
`- wagwest`	is the Internet HostName of the Unisys DCP or HLC

E.1.2 INPUT STOPPED BY MSG-WAIT

This message is displayed when you depressed MSG-WAIT in order to stop the data flow coming from the host. Your next transmission (XMIT key) will restart the host transmissions and this message will disappear.

E.1.3 => Connect to HOST FAILED

This message may be displayed in two cases:

1. At WE-UTS start-up time. If all parameters (including Application_Name and Mainframe_Host) where found in the configuration files, a first connection is attempted. If the “Application_Name” or “Term_Name” is invalid, you will get this message.

2. After a $$OPEN command, if the “Application_Name” is invalid.

E.1.4 => Connect CLOSED by HOST

This message is displayed when the Unisys host closed the connection (upon request or automatically after a time-out occurred). You can restart the same connection by entering a $$OPEN command without parameters.

E.1.5 Write ERROR to HOST => CLOSE

The last data couldn't be transmitted to the Unisys host. If this message occurs, the communication to the Unisys DCP was interrupted and you could try a $$OPEN command. If this does not work you get a message `(Socket to HOST FAILED!)`, this probably indicates a malfunction of the Unisys DCP (HLC) or a server configuration error of the WE-UTS. Let a system administrator check the following:

- Proper functioning of the Unisys DCP or HLC (TCP/IP software running?).
- Proper entries in the UNIX "services" and "hosts" files.

E.1.6 ERROR, NO HOST connection

This message is displayed, if you try a transmission (XMIT key) and the message `(- Connected to Application -)` is not displayed on your window status line. You should enter a $$OPEN command as probably requested in the upper border of your terminal window.

E.1.7 $ERROR: missing parameter(s)

This message is displayed if you try $$OPEN and either one of the "Application_Name" or "Mainframe_Host" is missing. Try again by providing both parameters in your $$OPEN command.

E.1.8 $CLOSE completed normally

This is the message you get after a $$CLOSE command. Simultaneously, an invitation to enter a $$OPEN command appears on the upper border of your window.

E.2 Configuration error messages displayed in the communication status field

Most configuration errors (i.e.: invalid fonts, missing parameters, etc.) will disallow WE-UTS to start (**no** terminal window at all) and you will get an error message on the console or on your shell window. However, some configuration errors can only be detected when the first connection to the Unisys host (through the DCP) is attempted. One of the next three messages can then be displayed:

E.2.1 => CONFIG Err:Service NOT FOUND

Means that the "Mainframe_Service" entry in the WE-UTS configuration file(s) have no counterpart in the UNIX "/etc/services" (or YP) file. You will have to correct that before retrying.

E.2.2 => CONFIG Err:Host NOT FOUND !

Means that the "Mainframe_Host" entry in the WE-UTS configuration file(s) or the one given with your $$OPEN command have no counterpart in the UNIX "/etc/hosts" (or YP) file. You will have to correct that before retrying.

E.2.3 => Socket to HOST FAILED !

Means that the “Mainframe_Service” or the “Mainframe_Host” entries in the WE-UTS configuration file(s) have been found in “/etc/services” and “/etc/hosts” (or YP) but that a “socket” of the proper type couldn’t be created. Check with your UNIX system administrator.

E.3 Fatal error messages displayed in the communication status field

E.3.1 => WE-COMD EXIT WITH CODE x !!

If this message appears, some abnormal conditions occurred in the “WE-COMD” communication server. You will have to close this WE-UTS window and reopen a new one to continue your work. This will also affect other WE-UTS emulation connected to the same WE-COMD server.

E.4 Layout of the WE-UTS upper window border for DCP TELNET connections

The following messages can be displayed in the WE-UTS upper window border for a DCP TELNET connection:

a) `ENTER: $$OPEN ApplName, [SystName]`

This message is displayed in three main cases:

1. At start-up time if any or both of the parameters (Appl_Name / Mainframe_Host) were missing to try a connection.
2. After a $$CLOSE command, this field contains the old ApplName and SystName. If you type a new $$OPEN command without arguments, you will be connected to the same application on the same system as the $$CLOSE was before.
3. After any event which closed the connection to the Unisys DCP.

In any case, a $$OPEN command will allow you to start (restart) your session with the Unisys host.

b) `Term_Name: Session <Appl_Name> on system <Mainframe_Host>`

This message is displayed at all times when your session with the application has been successfully established.

E.5 Layout of the WE-UTS upper window border for HLC Tp0 connections

The following messages can be displayed in the WE-UTS upper window border for a HLC Tp0 connection:

a) `ENTER: $$OPEN ApplName, [Appl_Server], [SystName]`

This message is displayed in three main cases:

1. At start-up time if any of the parameters (Appl_Name / Appl_Server / Mainframe_Host) were missing to try a connection.
2. After a $$CLOSE command, this field contains the old ApplName, Appl_Server and SystName. If you type a new $$OPEN without arguments, you will be connected to the same application using the same application server on the same mainframe as before.
3. After any event which closed the connection to the Unisys HLC.

In any case, a $$OPEN command will allow you to start (restart) your session with the Mainframe_Host.

b) `Term_Name: Session <Appl_Name> on system <Mainframe_Host>`

This message is displayed at all times when your session with the application has been successfully established.

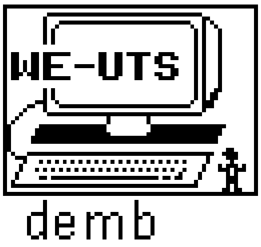
E.6 Content of the WE-UTS icon

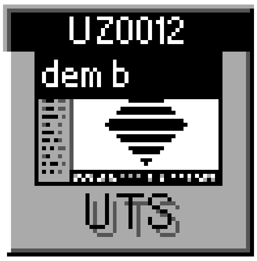
The WE-UTS icon can take two aspects depending on an active connection with the Unisys DCP (HLC).

1. If the connection is active:

The WE-UTS icon contains the Appl_Name to which this WE-UTS is connected.

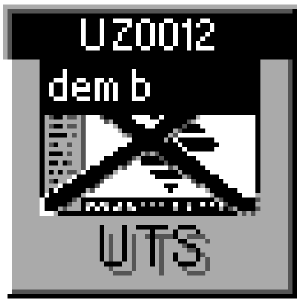
.

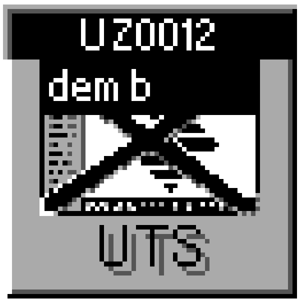




2. If no connection is active:

   The WE-UTS icon takes the shape of a tombstone or the Appl_Name of the last application is displayed.





F “M_func_names” list

A list of the names that can be used in the optional “Buttons_File”:

F.1 Send data (as defined by UNISYS™) to the host

M_XMIT	XMIT data to the host
M_MSG_WAIT	Message wait to the host
M_F1	F1 to the host
M_F2	F2 to the host
M_F3	F3 to the host
M_F4	F4 to the host
M_F5	F5 to the host
M_F6	F6 to the host
M_F7	F7 to the host
M_F8	F8 to the host
M_F9	F9 to the host
M_F10	F10 to the host
M_F11	F11 to the host
M_F12	F12 to the host
M_F13	F13 to the host
M_F14	F14 to the host
M_F15	F15 to the host
M_F16	F16 to the host
M_F17	F17 to the host
M_F18	F18 to the host
M_F19	F19 to the host
M_F20	F20 to the host
M_F21	F21 to the host
M_F22	F22 to the host

F.2 Send data (depending on macro content) to the host

M_MACRO1EXEC	Execute macro 1
M_MACRO2EXEC	Execute macro 2
M_MACRO3EXEC	Execute macro 3
M_MACRO4EXEC	Execute macro 4
M_MACRO5EXEC	Execute macro 5
M_MACRO6EXEC	Execute macro 6
M_MACRO7EXEC	Execute macro 7
M_MACRO8EXEC	Execute macro 8
M_MACRO9EXEC	Execute macro 9
M_MACRO10EXEC	Execute macro 10
M_MACRO11EXEC	Execute macro 11
M_MACRO12EXEC	Execute macro 12

F.3 Functions to move the cursor

M_HOME	Go to home position
M_UP	Move cursor 1 position upward
M_LEFT	Move cursor 1 position left
M_RIGHT	Move cursor 1 position right
M_DOWN	Move cursor 1 position downward
M_LINE_DUP	Duplify a line
M_TAB_FORW	Tab forward
M_TAB_BACK	Tab backward
M_RETURN	Go to next field on next line
M_BACK_SPACE	Go 1 position left and delete 1 char

F.4 Local editing functions

M_ERASE_DISP	Will erase whole display
M_LINE_DEL	Delete 1 line
M_ERASE_EOF	Erase to end of field
M_ERASE_EOD	Erase to end of display
M_ERASE_EOL	Erase to end of line
M_CHAR_ERASE	Erase character and move cursor
M_DEL_INDISP	Delete 1 character in display
M_DEL_INLINE	Delete 1 character in line
M_INS_INDISP	Insert 1 character in display
M_INS_INLINE	Insert 1 character in line
M_LINE_INS	Insert 1 line
M_TAB_SET	Set a tab at cursor position
M_SOE	Enter an SOE at cursor position
M_UNLOCK	Unlock keyboard
M_PRINT	Execute a hardcopy of the screen
M_COPY	Copy present selection to pasteboard
M_PASTE	Copy pasteboard to cursor position

Index